TECHNICAL UNIVERSITY OF CRETE – DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

A Review and Recommendation Tool for Restaurant Clients and Owners

Petros Pikionis

A thesis presented to the Technical University of Crete in fulfillment of the thesis requirement for the degree of Bachelor in Electronic and Computer Engineering

10/8/2020

Acknowledgements

I would like to thank my thesis supervisor Prof. Antonios Deligiannakis, for his supervision, support and advice. I would also like to thank my thesis readers Prof. Chalkiadakis Georgios and Mania Aikaterini.
Furthermore, I would like to thank some other people who contributed to the successful completion of this thesis. For their valuable advice and continuous support I would like to thank the members of the Distributed Information Systems and Applications Lab (MUSIC) of the Technical University of Crete, and especially Nektarios Moumoutzis, for his invaluable daily help, advice and support.
Last but not least, I would like to thank my parents for their never ending love and support, and their patience with me especially over the past few months. I dedicate this thesis to them.

Petros Pikionis

Technical University of Crete

September 2020

Table of Contents

List of Figures

Abstract

We describe the design and implementation of a web application, which provides insight and information to business owners that belong in the food and beverage sector. This is achieved by means of questionnaires given to customers of such businesses. The feedback provided by customers involves the location and service quality of a business, the price-to-quality ratio, as well as their opinion on dishes/beverages that are unique to said business. While many other competing applications in the FnB sector are able to provide data in the form of customer reviews, they do so in an unorganised manner, and ultimately fail to be of much use to business owners. In my opinion, the first step to improve a business through customer feedback is to dissect the business in its core features, such as the aforementioned quality of food, quality of service, etc. Afterwards, there needs to be a system of questionnaires designed to target each feature separately, so that the data extracted can be translated into statistics and graphs in a more accurate and digestible way. Ultimately, the development of our application strongly focuses on these two points.

Making use of our application begins by coming in contact with a business owner, in order to better understand what his business's unique selling points are. Building on this information provided, a set of questionnaires is created for each selling point. After consulting with the business owner, we decide on the different products that will be included in our application. Each product will be linked to a specially created questionnaire component. The final questionnaire given to the customer takes its form by combining all the questionnaire components relative to his dining experience. In practice, when a waiter takes an order, data is sent to our application, which takes upon the task of figuring out if there are any products in the order linked to questions, and a final questionnaire is created. Through this process, business owners will have a mapped out illustration of customer feedback in their hands. The information provided takes its form using multiple graphical techniques, such as histograms, pie charts etc.

This approach to customer reviews is what makes our application unique and superior to other software option

s. Most tools or applications limit their functionality to just asking for a rating or to leaving a basic review of their dining experience. Visible improvement in the qualities and services of a restaurant necessitates the proper collection and processing of the data provided by customer feedback. Improvement is a multi-step process that involves communication between customer and business owner. This communication guides improvements of the customer experience and can empower positive change in any business - even (and especially) when it's negative. This is the fundamental principle behind customer feedback.Ultimately, our application will also have the ability to interact with customers in response to the review they left.

Περιγράφουμε το σχεδιασμό και την εφαρμογή μιας διαδικτυακής εφαρμογής, η οποία παρέχει πληροφορίες και πληροφορίες σε ιδιοκτήτες επιχειρήσεων που ανήκουν στον τομέα των τροφίμων και των ποτών. Αυτό επιτυγχάνεται μέσω ερωτηματολογίων που δίνονται σε πελάτες τέτοιων επιχειρήσεων. Τα σχόλια που παρέχονται από τους πελάτες περιλαμβάνουν την τοποθεσία και την ποιότητα υπηρεσίας μιας επιχείρησης, την αναλογία τιμής προς ποιότητα, καθώς και τη γνώμη τους για πιάτα / ποτά που είναι μοναδικά για την εν λόγω επιχείρηση. Ενώ πολλές άλλες ανταγωνιστικές εφαρμογές στον τομέα FnB είναι σε θέση να παρέχουν δεδομένα με τη μορφή αξιολογήσεων πελατών, το κάνουν με μη οργανωμένο τρόπο, και τελικά αποτυγχάνουν να είναι πολύ χρήσιμα για τους ιδιοκτήτες επιχειρήσεων.
Κατά τη γνώμη μου, το πρώτο βήμα για τη βελτίωση μιας επιχείρησης μέσω της ανατροφοδότησης των πελατών είναι η αποσύνθεση της επιχείρησης στα βασικά χαρακτηριστικά της, όπως η προαναφερθείσα ποιότητα φαγητού, η ποιότητα των υπηρεσιών κ.λπ. Στη συνέχεια, πρέπει να υπάρχει ένα σύστημα ερωτηματολογίων σχεδιασμένο να στοχεύει σε κάθε χαρακτηριστικό ξεχωριστά, έτσι ώστε τα δεδομένα που εξάγονται να μπορούν να μεταφραστούν σε στατιστικά γραφήματα με πιο ακριβή και εύπεπτο τρόπο. Τελικά, η ανάπτυξη της εφαρμογής μας εστιάζει έντονα σε αυτά τα δύο σημεία.
Η χρήση της εφαρμογής μας ξεκινά έρχεται σε επαφή με έναν ιδιοκτήτη επιχείρησης, προκειμένου να κατανοήσει καλύτερα ποια είναι τα μοναδικά σημεία πώλησης της επιχείρησής του. Με βάση αυτές τις πληροφορίες, δημιουργείται ένα σύνολο ερωτηματολογίων για κάθε σημείο πώλησης. Μετά από διαβούλευση με τον ιδιοκτήτη της επιχείρησης, αποφασίζουμε για τα διαφορετικά προϊόντα που θα περιληφθούν στο ερωτηματολόγιο του.. Κάθε προϊόν θα συνδέεται με ένα ειδικά δημιουργημένο ερωτηματολόγιο. Το τελικό ερωτηματολόγιο που δίνεται στον πελάτη συνδυάζει όλα τα στοιχεία του ερωτηματολογίου σε σχέση με την εμπειρία του φαγητού. Στην πράξη, όταν ένας σερβιτόρος λαμβάνει μια παραγγελία, τα δεδομένα αποστέλλονται στην εφαρμογή μας, η οποία αναλαμβάνει το καθήκον να εξακριβώσει εάν υπάρχουν προϊόντα στην παραγγελία που συνδέονται με ερωτήσεις και δημιουργείται ένα τελικό ερωτηματολόγιο. Μέσω αυτής της διαδικασίας, οι ιδιοκτήτες επιχειρήσεων θα έχουν μια χαρτογραφημένη απεικόνιση των σχολίων των πελατών στα χέρια τους. Οι παρεχόμενες πληροφορίες έχουν τη μορφή τους χρησιμοποιώντας πολλαπλές γραφικές τεχνικές, όπως ιστογράμματα, γραφήματα πίτας κ.λπ.
Αυτή η προσέγγιση στις κριτικές πελατών είναι αυτό που κάνει την εφαρμογή μας μοναδική και ανώτερη από άλλες επιλογές λογισμικού. Τα περισσότερα εργαλεία ή εφαρμογές περιορίζουν τη λειτουργικότητά τους στο να ζητάτε μόνο μια βαθμολογία ή να αφήνετε μια βασική κριτική για την εμπειρία φαγητού. Η ορατή βελτίωση στις ποιότητες και τις υπηρεσίες ενός εστιατορίου

απαιτεί τη σωστή συλλογή και επεξεργασία των δεδομένων που παρέχονται από τα σχόλια των πελατών. Η βελτίωση είναι μια διαδικασία πολλαπλών βημάτων που περιλαμβάνει επικοινωνία μεταξύ πελάτη και ιδιοκτήτη επιχείρησης. Αυτή η επικοινωνία καθοδηγεί βελτιώσεις της εμπειρίας των πελατών και μπορεί να ενισχύσει τη θετική αλλαγή σε οποιαδήποτε επιχείρηση - ακόμη και (και ειδικά) όταν είναι αρνητική. Αυτή είναι η θεμελιώδης αρχή πίσω από τα σχόλια των πελατών. Τελικά, η εφαρμογή μας θα έχει επίσης τη δυνατότητα να αλληλεπιδρά με πελάτες ως απάντηση στην κριτική που άφησαν.

Chapter 1 - Introduction and objectives

1.1 Organization of the Thesis

Chapter 1 introduces the idea behind the application and lists the objectives and contribution of our Thesis. Studies have suggested that the implementation of a better structured space with analysis of statistics in the food and beverage space is considered necessary. The chapter introduced major viewpoints, which form the basis of the research.
Chapter 2 describes the research conducted on related technologies and services in the industry. and presented the tools that exist now in the market and the effort of our own research to find better and more efficient solutions.
Chapter 3 introduces the concept of Personas and how they interact with our application, and presents Use Cases. Chapter 3 introduced the case studies that were concerned with the implementation and the organization level of each one user.
Chapter 4 analyzes the Web System Architecture and Implementation Technologies of our application. We describe the architectural model that our communication model is built on.
Chapter 5 describes the Database Design Implementation and Database Interface. We analyze the E-R diagrams one by one for each class and at the end we present the whole connection between the entities.
Chapter 6 describes the directory Architecture and analyzes the files we used to implement our application. In this chapter we describe the design architecture of our application, we analyse its components and provide an in depth analysis of its operational methods.
Chapter 7 presents the User Interface of the application. In this chapter we describe the basic principles we followed in designing the User Interface.
Chapter 8 gives a description of future implementation plans and a summary of the Thesis. This section discusses the areas in which the developed system can be improved. These points mainly concern its expansion in terms of cooperation.

1.2 Introduction

For several years now the development of the IT industry has been primarily concerned with web applications as its main target of focus. The evolution of web applications has shown us their transformation from simple site presentations to complex architects providing functional tools to small businesses. Every organization in recent times has necessitated the use of web applications for updating employees with targets, aims, future plans as well as being a tool to be used for the everyday processes with the organization, and finally serving as an intermediate tool with which to market and communicate with external bodies and final users. Mobile phones have become the primary digital device in use due to major improvements in their hardware capabilities - i.e. high CPU power, better internet connectivity, bigger storage capacity and their sense of the external context. They are no longer an option but a necessity. Prediction says that in 2020, mobile phones will overtake PCs by far as the most common Web access device worldwide. According to our research, the total number of PCs in use reached 1.34 billion units in 2019. By 2019, the installed smartphones exceeded 3.5 billion units. Deloitte and Touche showed in their research that customer-centric companies had 60% more profits than non-customer-centric companies. As a result, FnB businesses should expect greater results by using customer feedback tools as a first step rather than anything else. In modern society, receiving the expected good quality of products in the food and beverage business is more important than ever before. Currently, there is no available digital map tool, with which one can explore an area for its restaurants in conjunction with ratings of their dishes and products. During the tourist season, business owners are overburdened by the increased amount of work on their shoulders, and as a result, are not in a suitable frame of mind to see the errors in their business, nor are they able to carefully reassess and restructure their business plan for the next season. The error of their judgement is amplified due to how unwelcome face to face feedback is perceived, as most customers struggle to report on any flaws they found in their dining experience. The addition of digital questionnaires eliminates confrontation and improves the accuracy of the feedback reported. Higher standards and expectations from customers have made businesses step up and meet these needs in exchange for their loyalty. The customer loyalty function has the "customer-centricity" philosophy as its main argument. This necessitates the study and optimisation of the reviewing process. The next level in customer feedback would allow diners to leave a review there and then, so as to preserve the accuracy of their impressions. Through our application, customers will receive the notion that the restaurant they're visiting respects their sentiments and has a vested interest in their opinion, which will ultimately give credence to the business. Our application in the future will be a guidance to a taste experience based on the dishes that each store has to offer.

1.3 The objectives of the Thesis

1. To examine the prerequisites for offering critical information related to the business spatial information related to food, beverages and services offered.

2. To analyze the requirements for supporting business individuals, business communities and external users with personalized spatiotemporal management of the questionnaires.

3. To design functionality and interfaces for offering these services on top of web devices.

4. To explore standards, technologies, tools and architectures for offering this kind of rich multimedia functionality in diverse platforms utilizing a web application approach supported by emerging standards.

5. To evaluate the application with real users, as well as the design and technological decisions taken, and the state of the technologies and tools used as well as their adoption, and propose future research and development efforts.

6. The implementation of an application with a user-friendly User Interface that integrates a Form Builder that dynamically creates questionnaires, personalised for each session and each user. The results of the questionnaires are concentrated, are analysed and presented in the form of statistical graphs.

1.4 Thesis Contribution

In this dissertation, we show that the effective use of data from questionnaires is based on a set of performance parameters, which eloquently delineate the performance capabilities of the target communication system. This communication model is a performance understanding tool, which is based on a system of dynamic personalized questionnaires and records performance characteristics, thus facilitating business executives to analyze and maximize their future business plan. Through this set of performance parameters - the communication model, we can analyze, predict, evaluate and explain issues related to a high speed evaluation in the food and beverage business. This communication model is a tool that aims to understand and improve performance. The model is based on a visually interactive engagement during a potential customer on a business site.
The main contributions of this dissertation are:
We present a realistic communication model with the construction of the application that supports the understanding of the performance of the cluster communication system. This model of communication becomes the foundation of this research work. The communication model is presented in detail and gives us a complete analysis of the ultimate goal that the application wants to have and to be able to produce data through the catering shops. The other process described in our dissertation is the development and implementation of a system for creating questionnaires - questions as well as auxiliary tools for the user of our application. Based on the architectural model of our system, we have developed an effective communication program for full exchange between user and business. The effective implementation and use of the tools of

our application will be able to lead to a level of operation that will deliver precision data in areas with intense activity in the field of catering. If our communication model is used effectively we will be able to achieve a mapped area based on products, services and special characteristics of each business.

Chapter 2: Related Research and Contribution of our Application

2.1 Related Research

During the design and development of our Application, research was conducted on systems and applications that support the food and beverage business. The systems identified incorporate desirable features in the direction of our application, but only support parts of our applications.

Survey.js

Survey.js the most feature-rich Survey / Form Library available at the current moment – SurveyJS is a modern way to add surveys and forms to your website. It can be easily customized and extended to suit your needs. In the early development stages of our application, Survey.js presented itself as the obvious choice to integrate into our system, to function as a form builder. However, our systems architecture would necessitate a separate installation of our application to each new business we would affiliate with, through which we would achieve a safer, smaller and more manageable database. This effect in conjunction with the yearly subscription costs of Survey.js would escalate the cost of utilising it in our system, making it an unaffordable choice for our business model.

In this thesis we chose to build our own specialised form builder, suited for our applications needs. This specialised form builder is, in essence, a set of small components, and each component corresponds to a type of question - a more in depth explanation is given in the next chapters. An added benefit is the disengagement from any kind of policies or update issues that might occur.

Google forms

Google Forms is a form tool that comes free with your Google account. You can drag and drop questions in the order you like, add standard questions type, gather responses in forms or save them to a Google Sheets spreadsheet as well as customize the form with simple photo or color themes. It boasts a plethora of customisation options and settings choices, as well as the option to install add-ons for your Forms. Google Forms is a tool that enables the collection of user information in the form of a personalized survey or quiz. The data collected is automatically parsed into a spreadsheet. We were disheartened to use Google Forms due to a set of reasons:

Firstly, we opted out of using a tool we would have a dependency on. A faulty update could endanger the integrity of our application. Secondly, compatibility issues with certain functions of our application discouraged us from using Google Forms. Finally, data privacy is more important

than ever and should not be neglected as a factor. Google is notorious for manipulating user data and as such the creation of our own Form Builder is even more so justified.
In this thesis, the development of our own specialised form builder solves all issues we would possibly encounter by using 3rd party software tools - at the cost of only minimal development time.

Trip Advisor

TripAdvisor.com is a travel-focus platform where one can acquire travel information by reading reviews and opinions of content related to travelling. The website also includes forums where users can discuss and communicate with each other. TripAdvisor boasts being a pioneer in having the platform's contents created by its own users. The website provides its content free of charge to its users. TripAdvisor hosts digital spaces for businesses that sign up for it, and in that space a visitor can find information regarding the location, amenities, price ranges and media content among other features.
Most importantly, reviews in the form of a public comment can be left by a visiting customer, as well as a basic zero-to-five star rating. Through this model of presentation and the gravity tripAdvisor places on its simplistic reviewing system, a loop of information between the user and the website is created, that leaves the business uninvolved in the process. In summary, TripAdvisor's business model stands on taking advantage of a system of abstract reviews left by users to be read by potential future customers.
In this thesis, compared to TripAdvisor, our application is built on a different mentality around customer feedback. In our research we concluded that questionnaires are the most powerful means of measuring customer perceptions. We went a step further and created a mechanism that creates real time dynamic questionnaires, based on the order given by the customer. Furthermore, this method allows the business owner to have an active role in the process of data collection, for instance by selecting which products are worth evaluating. A keynote to take into consideration is the fact that, in its backbone, our application is not a competitive counter choice to TripAdvisor. We take a different approach to collecting and analyzing customer feedback. Our philosophy revolves around collecting feedback that contains information in a verbatim format and is characterized by a higher level of detail, describing the most critical elements for customer experience, whereas TripAdvisor prefers a more unstructured reviewing system, in which users can describe their experiences more freely, but without substance.
Despite all this, the future finds our mobile application having elements of advertising, making it a competitive alternative to TripAdvisor on the market. One such element is a method of promoting businesses that affiliate with us. User preferences will filter through all the possible choices, either actively in the form of a search field, or indirectly by extrapolating the data stored by their history of answered questionnaires, ultimately advertising the restaurant they will most likely be interested in.

2.2 Summary

In the second Chapter we describe the process we used to try to calculate the opportunity success rate of our application. In other words we speculate what place our implementation might hold in the Food and Beverage industry. A description of the most apparent competitive platforms and services is given, followed by reasons they are not. We found Google Forms and Survey.js too limiting for the implementation of our idea, and as such we created our own Form Builder, that enables our application to have a competitive edge in the Food and Beverage Industry compared to a platform like TripAdvisor or any other similar service.

Chapter 3 - Requirements Analysis, User Interface Prototyping and Evaluation

3.1 Introduction

Software development requires the step by step cooperation of the design and development team. These steps ensure that the final product not only is fully functional but also provides a user friendly interface. These steps also take into account the time and resource constraints that will emerge. The analysis of user groups and user needs will define the end goals and requirements to be kept in track for the development design. More specifically, once designers know what the product is about, what are its uses and who the target users are, they can start designing their product interface to be optimized, with functionality and ease of use in mind. After interface and design optimizations have been finalized and the application logic has been decided the development process can begin. Throughout this process all steps need constant evaluation. An abundance of evaluations in the early stages of design will result in a much cheaper troubleshooting cost and an easier facilitation of future changes. In this Chapter, a description of the process of requirement analysis and prototyping is given. The tools that we used are Personas definition, Use Cases, Storyboarding,Paper Prototyping and a Market Research. During the phases of prototyping we used Adobe XD for designing the interface of our web application.

3.2 Requirements for Functionality and Interfaces

This application is a "swiss knife" of feedback collection for the business owner, including but not restricted to product and customer service reviews, while making the statistical analysis faster, targeted and easier to access. The application will allow you to display questions depending on what the customer ordered, questions will be about quality, service, price, location as well as questions that will be displayed accordingly with the answers given by the customers. The application will give us the ability to create our personalised questions, integrate them into questionnaires and distribute them for use in our business.In the first stage of the development the questionnaire will be delivered via a tablet, at a later stage of the development customer end phone application integrated. The store owner will have access to customer responses and graphs- statistics analysis that will come from the questions.

3.3 Personas

Personas are fake users that assist in the profiling of real world users. By defining user personas, the designer comprehends the unique user needs and can help achieve the goals set by the business owner. In the process of creating the potential application users we included 4 types of people that interact with the application. These include the panel administrator, a restaurant owner, an employee and a customer.

3.3.1 Petros, 29, admin

Petros is the panel administrator. He is in contact with a potential business manager/owner to create the questionnaires and determine the products to be scored. Petros automated emails with the statistics and graphs for each question to the business manager/owner.

3.3.2 John, 42, restaurant owner

John, in collaboration with Petros, will decide upon the questionnaire content and what the store's questions will be, what dish should be scored, what he would like to see in the statistics and graphs derived from each question. The application provides John the ability to see in real time how his store is rated.

3.3.3 Jane, 18, employee

Jane is a waitress. After receiving the order from the guest Jane enters the order in our application, which creates a personal questionnaire for each customer which is provided via a tablet or smartphone.

3.3.3 Dimitris, 29 , guest

Dimitris is a guest who has ordered three dishes. At the end of his stay, Jane asks if he would like to give any feedback on his experience at the restaurant . Dimitris accepts and is given a tablet/smartphone to fill out a questionnaire.

3.5 Use Cases

Use Case 1	The Administrator, manager or user login	
Goal In Context	Administrator, manager or user wants to login to the application. The application must ensure that the administrator, manager or user is securely connected.	
Scope & Level	Management and safe login of users .	
Preconditions	Administrator has already registered those who will have access to the application.	
Success End Condition	Administrator, manager or user has successfully logged in to the application.	
Failed End Condition	Administrator, manager or user has failed to login to the application.	
Primary, Secondary Actors	Users / System management of User	
Trigger	Administrator, manager or user wants to login to the application	
Description	**Step**	**Action**
	1	The system displays the login form.
	2	Administrator, manager or user completes the login form.
	3	The system checks if the values match the database.
	5	The system displays the main page(dashboard) of the application.
Extensions	**Step**	**Branching Action**
	3 α	Administrator, manager or user hasn’t completed his login form correctly. 3 α .1 The system suggests completing the form again by step 2.

Use Case 2	The administrator, manager or user logout	
Goal In Context	Administrator, manager or user wants to logout from the application.	
Scope & Level	Management and safe logout of users .	
Preconditions	Administrator, manager or user has already visited the application	
Success End Condition	Administrator, manager or user has successfully logged out from the application.	
Failed End Condition	Administrator, manager or user has failed to logout from the application.	
Primary, Secondary Actors	User / System management of User	
Trigger	Administrator, manager or user wants to logout from the application	
Description	**Step**	**Action**
	1	The system displays the logout button
	2	Administrator, manager or user agrees to logout from the application.
	3	The system displays the login page.
Extensions	**Step**	**Branching Action**

Use Case 3	The administrator, manager or user browses the dashboard menu

Goal In Context	Admin wants to display the page "Dashboard menu " .	
Scope & Level	Part of the application " Dashboard menu ".	
Preconditions	Admin has already visited the application	
Success End Condition	Admin has successfully logged in to the application.	
Failed End Condition	Admin has failed to register to the application.	
Primary, Secondary Actors	Users / System management of Users	
Trigger	The Admin wants to enter in the application	
Description	**Step**	**Action**
	1	The system displays the page " Manage Users "
Extensions	**Step**	**Branching Action**

Use Case 4	The administrator or manager browses the page for Users
Goal In Context	Admin wants to display the page of " Manage Users " .
Scope & Level	Part of application " Manage Users ".
Preconditions	Admin has already visited the application
Success End Condition	Admin has successfully logged in to the application.
Failed End Condition	Admin has failed to register to the application.

Primary, Secondary Actors	User / System management of Users	
Trigger	Admins wants to enter in the page for “ Manage Users “	
Description	**Step**	**Action**
	1	The system displays the page “ Manage Users “
Extensions	**Step**	**Branching Action**

Use Case 5	The administrator adding an user	
Goal In Context	Admin wants to add another user in the application.	
Scope & Level	Users	
Preconditions		
Success End Condition	1. The admin has successfully logged in to the application. 2. The admin has entered the Page “Users“.	
Failed End Condition	Admin has failed to add an user.	
Primary, Secondary Actors	User / System management of User	
Trigger	The admin wants to add an user to the application	
Description	**Step**	**Action**
	1	Admin press the button “Add user”.
	2	The system displays a form to create the user.

	3	Admin adding name, email and password for the user.
	4	The system checks if the values are correct.
	5	The system checks if the values are unique.
	6	The system creates a new user and saves him in the database.
Extensions	**Step**	**Branching Action**
	4	Admin hasn’t completed the registration form. 4 α .1 The system informs the admin and suggests completing the form from step 2.
	5	The name or the email is not unique. The system informs the admin that the name or email has been used by another user and needed to complete the form from step 2.

Use Case 6	The Administrator editing an user
Goal In Context	Admin wants to edit an user in the application.
Scope & Level	Users
Preconditions	Admin has already visited the application
Success End Condition	1. Admin has entered the Page “Users“. 2. Admin has already added the new user.
Failed End Condition	Admin has failed to edit a user .
Primary, Secondary Actors	User / System management of User
Trigger	Admin wants to edit a user to the application

Description	**Step**	**Action**
	1	Admin presses the button “Edit user” .
	2	The system displays a form to edit the user.
	3	The admin adds the name, email and password for the user.
	4	The system checks if the values are correct.
	5	The system checks if the values are unique.
Extensions	**Step**	**Branching Action**
	4	Admin hasn’t completed the editing form. The system informs the user and suggests completing the form from step 2.
	4	The name or the email is not unique.The system informs the user that name or email has been used again and needs to complete the form from step 2.

Use Case 7	The administrator upgrades a manager to user
Goal In Context	Admin wants to upgrade a user to “manager”.
Scope & Level	Users
Preconditions	Admin has already visited the application
Success End Condition	Admin has entered the Page “Users“.
Failed End Condition	

Primary, Secondary Actors	User / System management of User	
Trigger	Admin wants to upgrade the state of user from “ user “ to “ manager “	
Description	**Step**	**Action**
	1	Admin presses the “ upgrade user ” .
	2	The system changes the state of user to “ manager “
	3	The system changes the access of managers to all levels .
Extensions	**Step**	**Branching Action**

Use Case 8	The administrator downgrades a manager to user
Goal In Context	Admin wants to downgrade a manager to “ user ”.
Scope & Level	Users
Preconditions	Admin has already visited the application
Success End Condition	1. Admin has entered the Page “Users“.
Failed End Condition	

Primary, Secondary Actors	User / System management of User	
Trigger	Admin wants to downgrade the state of user from “ manager “ to “ user “	
Description	**Step**	**Action**
	1	Admin presses the “ downgrade manager ” .
	2	The system changes the state of “ manager ” to “ user “
	3	The system changes the access of users to all levels .
Extensions	**Step**	**Branching Action**

Use Case 9	The administrator deleting a user/manager
Goal In Context	Admin wants to delete a user/manager in the application.
Scope & Level	Manage Users
Preconditions	Admin has already visited the application
Success End Condition	1. Admin has entered the Page “Users“.

Failed End Condition		
Primary, Secondary Actors	User / System management of User	
Trigger	Admin wants to delete a user/manager to the application	
Description	**Step**	**Action**
	1	Admin presses the button “delete”.
	2	The system displays a form to ask if admin is sure.
	3	The admin presses the button to ensure the system for deletion.
	4	The system removes the user from the database.
Extensions	**Step**	**Branching Action**

Use Case 10	The administrator editing access to a user/manager
Goal In Context	Admin wants to delete a user/manager in the application.
Scope & Level	Manage Users
Preconditions	Admin has already visited the application
Success End Condition	2. Admin has entered the Page “Users“.
Failed End Condition	

Primary, Secondary Actors	User / System management of User	
Trigger	Admin wants to delete a user/manager to the application	
Description	**Step**	**Action**
	1	Admin presses the button “delete”.
	2	The system displays a form to ask if admin is sure.
	3	The admin presses the button to ensure the system for deletion.
	4	The system removes the user from the database.
Extensions	**Step**	**Branching Action**

Use Case 11	The administrator, manager or user change their password	
Goal In Context	The user wants to change his password.	
Scope & Level	Management and safe registration of users .	
Preconditions	The user has already visited the application	
Success End Condition	The user has successfully logged in to the application.	
Failed End Condition	The user has failed to login to the application.	
Primary, Secondary Actors	User / System management of User	
Trigger	The user wants to login to the application	
Description	**Step**	**Action**

	1	The user logins in the system and wants to change his password.
	2	The user enters the “ manage users panel”.
	3	The user presses the button to edit his profile.
	4	The system displays to edit your profile.
	5	The user confirms a new password.
Extensions	**Step**	**Branching Action**
	1	The user hasn’t logged in to the application. 5 α .1 The system displays a message to complete the form of login . 5a.2 If you fail to login, the system sends an email to change on a new password.
	5	The user used the same password . 5 α .1 The system informs the user to use a new password for his new password.

Use Case 12	The administrator or manager browses the page for Fieldsets
Goal In Context	Administrator or manager wants to display the page “Fieldsets”.
Scope & Level	Part of application “Fieldsets“
Preconditions	Administrator or manager has already visited the application.
Success End Condition	Administrator or manager has successfully logged in to the application.
Failed End Condition	Administrator or manager has failed to register for the application.
Primary, Secondary Actors	User / System management of Users

Trigger	The user wants to enter in the page for “ Fieldsets “	
Description	**Step**	**Action**
	1	The system displays the page “ Fieldsets “ with all the questionnaires we have made.
Extensions	**Step**	**Branching Action**

Use Case 13	The administrator or manager adding a fieldset.	
Goal In Context	Administrator or manager will be able to create a questionnaire in which they must give a name and a code as a url key	
Scope & Level	Fieldsets	
Preconditions	Admin or manager has already visited the application	
Success End Condition	1. Admin or manager has successfully registered for the application. 2. Admin or manager has entered the Page “Fieldsets“ . 3. Admin or manager presses to add a fieldset.	
Failed End Condition		
Primary, Secondary Actors	User / System management of User	
Trigger	Admin or manager adds a fieldset to the application	
Description	**Step**	**Action**

	1	The system displays a button to add a fieldset form.
	2	Admin or manager completes the form with name and slug.
	3	The system checks if the values are correct.
	4	The system checks if the values are unique.
	5	The system creates a new fieldset and saves it in the database.
Extensions	**Step**	**Branching Action**
	3 α, 4 α	The fieldset name and slug are not unique. 3 α The system informs the user and suggests completing the form again by step 2.

Use Case 14	The administrator or manager editing a fieldset
Goal In Context	Administrator or manager will be able to edit the name and the slug (we will use it as a url key) for the questionnaire.
Scope & Level	Fieldsets
Preconditions	Administrator or manager has already visited the application
Success End Condition	1. Administrator or manager has successfully registered for the application. 2. Administrator or manager has entered the Page “Fieldsets“. 3. Administrator or manager has already created a questionnaire. 4. Administrator or manager presses to edit the questionnaire.

Failed End Condition		
Primary, Secondary Actors	User / System management of User	
Trigger	Admin or manager editing a fieldset to the application	
Description	**Step**	**Action**
	1	The system displays to edit a fieldset form.
	2	Administrator or manager completes the form with name and slug.
	3	The system checks if the values are correct.
	4	The system checks if the values are unique.
	5	The system re-writes the fieldset and saves it in the database.
Extensions	**Step**	**Branching Action**
	3 α , 4 α	The fieldset name and slug are not unique. 3 α The system informs the user and suggests completing the form again by step 2.

Use Case 15	The administrator or manager copying a fieldset.
Goal In Context	The administrator or manager wants to copy a created questionnaire and change a field or some parts of it without needing to recreate it from scratch.
Scope & Level	Fieldsets
Preconditions	The administrator or manager has already visited the application

Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Fieldsets“. 3. The administrator or manager has already created a questionnaire. 4. The administrator or manager presses to copy the already questionnaire.	
Failed End Condition		
Primary, Secondary Actors	User / System management of User	
Trigger	The administrator or manager copying a fieldset to the application	
Description	**Ste p**	**Action**
	1	The system displays the questionnaires
	2	The administrator or manager presses the button in one of the many questionnaires to copy it.
	3	The system creates the same questionnaire with different id.
Extensions	**Ste p**	**Branching Action**

Use Case 15	The administrator or manager puts a conditional field at the fieldset.
Goal In Context	The administrator or manager wants to add a condition to a specific questionnaire. The administrator picks the conditions and the linked question for his condition field.
Scope & Level	Fieldsets

Preconditions	The administrator or manager has already visited the application	
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Fieldsets“. 3. The administrator or manager has already created a questionnaire. 4. The administrator or manager presses to add a condition on an already Fieldset. 5. The administrator or manager picks the conditions and the linked question.	
Failed End Condition		
Primary, Secondary Actors	User / System management of User	
Trigger	The administrator or manager adds a condition field to the application	
Description	**Ste p**	**Action**
	1	The system displays the questionnaires
	2	The administrator or manager presses the button in one of the many questionnaires to add a condition on it.
	3	The Administrator or manager picks the conditions and the linked question on the Form.
	4	The system creates a conditional linked question with another question of our choice.
Extensions	**Ste p**	**Branching Action**

Use Case 16	The administrator or manager deleting a fieldset	
Goal In Context	The administrator or manager wants to delete a questionnaire. If a questionnaire is deleted then all the questions that exist in it and together with the charts corresponding to the question are deleted.	
Scope & Level	Fieldsets	
Preconditions	The administrator or manager has already visited the application	
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Fieldsets“ . 3. The administrator or manager presses to delete a questionnaire.	
Failed End Condition		
Primary, Secondary Actors	User / System management of User	
Trigger	The administrator or manager deletes a questionnaire to the application	
Description	**Step**	**Action**
	1	The administrator or manager wants to delete a questionnaire
	2	Τhe system asks the user if he is sure.
	3	The administrator or manager confirms the deletion .
	4	The system deletes the questionnaire.
	5	The system deletes the questionnaire in the database .
Extensions	**Step**	**Branching Action**

Use Case 17	The administrator or manager browse the page Manage fields	
Goal In Context	The administrator or manager will browse the questions when they press the button “managing fields”. The application will display all the questions for this questionnaire.	
Scope & Level	Fields	
Preconditions	The administrator or manager has already visited the application	
Success End Condition	4. The administrator or manager has successfully registered for the application. 5. The administrator or manager has entered the Page “Fieldsets->Manage Fields“ .	
Failed End Condition		
Primary, Secondary Actors	Fieldsets->(button)Manage Fields / System management of Fieldsets->Manage Fields	
Trigger	The administrator or manager put fields by order in the application	
Description	**Step**	**Action**
	1	The administrator or manager wants to put by order the fields
Extensions	**Step**	**Branching Action**

Use Case 18	The administrator or manager adding a field	
Goal In Context	The administrator or manager wants to add a field.	
Scope & Level	Manage fields of a fieldset.	
Preconditions	The administrator or manager has already visited the application	
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Fieldsets“ . 3. The administrator or manager presses to manage fields of a specific fieldset. 4. The administrator or manager adds a field on that fieldset.	
Failed End Condition		
Primary, Secondary Actors	Fieldsets-Manage fields / System management of Fieldsets-Manage Fields	
Trigger	The administrator or manager adding a field.	
Description	**Step**	**Action**
	1	The system displays a form to build your field.
	2	The administrator or manager completes the label, the name, the validation, the field type, the class, the note, checks if the field is conditional and if it's to choose an option of what conditional it is.
	3	The administrator or manager saves his choices.
	4	The system checks if the values are correct.

	5	The system creates a new field and saves it in the database.
Extensions	**Ste p**	**Branching Action**
	4	The administrator or manager hasn’t completed the form right. The system informs the user and suggests completing the form by step 2.

Use Case 19	The administrator or manager editing a field
Goal In Context	The administrator or manager wants to edit a field.
Scope & Level	Manage fields of a fieldset.
Preconditions	The administrator or manager has already visited the application
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Fieldsets“ . 3. The administrator or manager presses to manage fields of a specific fieldset. 4. The administrator or manager has already created the field. 5. The administrator or manager edits the field on that fieldset.
Failed End Condition	
Primary, Secondary Actors	Fieldsets-Manage fields / System management of Fieldsets-Manage Fields
Trigger	The administrator or manager presses to add a field.

Description	Step	Action
	1	The system displays a form to edit your field.
	2	The administrator or manager fills in the label, the name, the validation, field type, class, note,checks if the field is conditional and if it's to choose the option of what conditional it is.
	3	The administrator or manager submits the choices.
	4	The system checks if the values are correct.
	5	The system creates a new field and saves it in the database.
Extensions	**Step**	**Branching Action**
	4	The administrator or manager hasn’t completed the form right. The system informs the user and suggests completing the form by step 2.

Use Case 20	The administrator or manager deleting a field
Goal In Context	The administrator or manager wants to delete a field.
Scope & Level	Manage fields of a fieldset.
Preconditions	The administrator or manager has already visited the application
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Fieldsets“ .

	3. The administrator or manager presses to manage fields of a specific fieldset. 4. The administrator or manager deletes a field on that fieldset.	
Failed End Condition		
Primary, Secondary Actors	Fieldsets-Manage fields / System management of Fieldsets-Manage Fields	
Trigger	The administrator or manager presses to delete the field.	
Description	**Step**	**Action**
	1	The administrator or manager wants to delete a field.
	2	Τhe system asks the user if he is sure.
	3	The administrator or manager confirms his deletion.
	4	The system deletes the field.
	5	The system deletes the field in the database .
Extensions	**Step**	**Branching Action**

Use Case 21	The administrator or manager browses the page for Forms
Goal In Context	The administrator or manager browses the page for Forms. At that part of the application we are creating profiles for each shop.
Scope & Level	Part of application “ Forms “
Preconditions	The administrator or manager has already visited the application

Success End Condition	The administrator or manager has successfully logged in to the application.	
Failed End Condition	The administrator or manager has failed to register to the application.	
Primary, Secondary Actors	Forms / System management of Users	
Trigger	The administrator or manager wants to enter in the page “ Forms “	
Description	**Step**	**Action**
	1	The system displays the page “ Forms “
Extensions	**Step**	**Branching Action**

Use Case 22	The administrator or manager adding a form	
Goal In Context	The administrator or manager wants to add a new profile shop for the manager.	
Scope & Level	Forms	
Preconditions	The administrator or manager has already visited the application	
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Forms“ . 3. The administrator or manager adds a new profile shop.	
Failed End Condition		
Primary, Secondary Actors	Forms / System management of Forms	
Trigger	The administrator or manager presses to add form.	
Description	**Step**	**Action**

	1	The system displays a page to create your form.
	2	The administrator or manager completes the fields: name , the slug, the associated questionnaire, the template, the logo, the redirect page and a submit message.
	3	The administrator or manager saves the choices.
	4	The system checks if the values are correct.
	5	The system creates a new form and saves it in the database.
Extensions	**Step**	**Branching Action**
	4	The administrator or manager hasn't completed the form right. The system informs the user and suggests completing the form by step 2.

Use Case 23	The administrator or manager editing a form
Goal In Context	The administrator or manager wants to edit a form.
Scope & Level	Forms
Preconditions	The administrator or manager has already visited the application
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the PageForms . 3. The administrator or manager has already created a form. 4. The administrator or manager edits the form.
Failed End Condition	
Primary, Secondary Actors	Forms / System management of Forms
Trigger	The administrator or manager presses to edit the form.

Description	Step	Action
	1	The system displays a page to edit your form.
	2	Admin or manager edits the name , the slug and the associated fieldset.
	3	The administrator or manager saves his choices.
	4	The system checks if the values are correct.
	5	The system updates and saves the form in the database.
Extensions	**Step**	**Branching Action**
	4	The administrator or manager hasn't completed the form right. The system informs the user and suggests completing the form by step 2.

Use Case 24	The administrator or manager deleting a form
Goal In Context	The administrator or manager wants to delete a form.
Scope & Level	Forms
Preconditions	The administrator or manager has already visited the application
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the PageForms . 3. The administrator or manager deletes a Form. 4. The system deletes all the graphs and statistics for that form.
Failed End Condition	
Primary, Secondary Actors	Forms / System management of Forms

Trigger	The administrator or manager deletes a form.	
Description	**Step**	**Action**
	1	The administrator or manager wants to delete a form.
	2	Τhe system asks the user if he is sure.
	3	The administrator or manager confirms his deletion.
	4	The system deletes the form.
	5	The system deletes the form in the database.
Extensions	**Step**	**Branching Action**

Use Case 25	The administrator or manager can view the survey
Goal In Context	The administrator or manager wants to view the survey.
Scope & Level	Form
Preconditions	The administrator or manager has already visited the application
Success End Condition	5. The administrator or manager has successfully registered for the application. 6. The administrator or manager has entered the Page Forms . 7. The administrator or manager looks at the completed Survey he made by clicking on the name of the Form.
Failed End Condition	

Primary, Secondary Actors	Forms / System management of Forms	
Trigger	The administrator or manager view the Survey	
Description	**Step**	**Action**
	1	The administrator or manager presses the button to preview the form with the questionnaire.
	2	The system displays the questionnaire in a new Page
Extensions	**Step**	**Branching Action**

Use Case 26	The Guest answering the survey
Goal In Context	The waiter brings the Survey in a tablet or phone and the guest answers the questions.
Scope & Level	Survey
Preconditions	The system has no information for the customer.
Success End Condition	1. The customer received our request to answer the Survey. 2. The customer linked with our Survey.

Failed End Condition		
Primary, Secondary Actors	Forms / System management of Forms	
Trigger	Guest answers the Survey	
Description	**Step**	**Action**
	1	The system displays the Survey.
	2	The guest answers all the necessary questions.
	3	The guest submits his answers.
	4	The system checks if the values are correct.
	5	The system save the answers in our database
Extensions	**Step**	**Branching Action**
	4	The guest hasn’t completed his answers. The system informs the customer and suggests completing the form by step 2.

Use Case 27	The administrator manager or user browses the page for Submission forms	
Goal In Context	The administrator, manager or user will display a list with all the answered submission Forms for the shop.	
Scope & Level	Submission Forms	
Preconditions	The administrator, manager or user has already visited the application	
Success End Condition	The administrator, manager or user has successfully logged in to the application.	
Failed End Condition	The administrator, manager or user has failed to register to the application.	
Primary, Secondary Actors	Submission forms / System management of Submission forms	
Trigger	The administrator, manager or user wants to enter in the page for “ Submission Form “	
Description	**Step**	**Action**
	1	The system displays a list with all the forms that exist and a number of how many answered it.
Extensions	**Step**	**Branching Action**

Use Case 28	The administrator, manager or user viewing the submission Forms
Goal In Context	The administrator, manager or user or User views the submission Forms
Scope & Level	Submission Forms
Preconditions	The administrator, manager or user has already visited the application

Success End Condition	1. The administrator, manager or user has successfully registered for the application. 2. The administrator, manager or user has entered the Page “Submission Forms“ . 3. The administrator, manager or user displays all the submission forms.	
Failed End Condition		
Primary, Secondary Actors	Forms / System management of Forms	
Trigger	The system displays all the submission forms.	
Description	**Step**	**Action**
	1	The system displays all the existing forms.
	2	The administrator, manager or user can choose to browse the submission forms by one of the all forms.
	3	The system displays all the submissions forms.
	4	The administrator, manager or user can choose to preview, print or delete a submission form.
Extensions	**Step**	**Branching Action**

Use Case 30	The administrator or manager browse the Form Emails	
Goal In Context	The administrator or manager wants to have a tool to send emails to their staff or customers.	
Scope & Level	Email Forms	
Preconditions	The administrator or manager has already visited the application	
Success End Condition	The administrator or manager has successfully logged in to the application.	
Failed End Condition	The administrator or manager has failed to register to the application.	
Primary, Secondary Actors	Emails / System management of Emails	
Trigger	The administrator or manager wants to enter in the page for " Form Emails "	
Description	**Step**	**Action**
	1	The system displays the page " Form Emails "
	2	The system browse all a list of emails Forms
Extensions	**Step**	**Branching Action**

Use Case 31	The administrator or manager adding an email form	
Goal In Context	The administrator or manager can add a new email form	
Scope & Level	Form emails	
Preconditions	The administrator or manager has already visited the application	
Success End Condition	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Form emails“ . 3. The administrator or manager now can add a new email form.	
Failed End Condition		
Primary, Secondary Actors	Email Form / System management of Emails	
Trigger	The administrator or manager now can add a new email form.	
Description	**Step**	**Action**
	1	The system displays a page to create a new email form.
	2	The administrator or manager fills in all the necessary blanks(Associated form,Type, Name, Subject, Sender, From, Message) in the email form.
	3	The administrator or manager saves his choices.
	4	The system checks if the values are correct.
	5	The system creates a new form and saves it in the database.
Extensions	**Step**	**Branching Action**
	4	The administrator or manager hasn’t completed the email form. The system informs the user and suggests completing the form by step 2.

Use Case 32	The administrator or manager Editing an email form	
Goal In Context	The administrator or manager can edit a new email form	
Scope & Level	The administrator or manager has already visited the application	
Preconditions	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Form emails“ . 3. The administrator or manager has already created the email form. 4. The administrator or manager can edit the email form now.	
Success End Condition		
Failed End Condition	Email Form / System management of Emails	
Primary, Secondary Actors	The administrator or manager can edit the email form.	
Trigger	**Step**	**Action**
Description	1	The system displays a page to edit the email form.
	2	The administrator or manager edits the necessary blanks in the email form.
	3	The administrator or manager saves his choices.

	4	The system checks if the values are correct.
	5	The system creates a new form and saves it in the database.
Extensions	**Step**	**Branching Action**
	4	The administrator or manager hasn't completed the email form. The system informs the user and suggests completing the form by step 2.

Use Case 33	The administrator or manager deleting an email form	
Goal In Context	The administrator or manager can deleting a new email form	
Scope & Level	The administrator or manager has already visited the application	
Preconditions	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page "Form emails" . 3. The administrator or manager has already created the email form. 4. The administrator or manager can delete the email form now.	
Success End Condition		
Failed End Condition	Email Form / System management of Form	
Primary, Secondary Actors	The administrator or manager can delete the email form.	
Trigger	**Step**	**Action**
Description	1	The administrator or manager wants to delete an email form.
	2	Τhe system asks the user if he is sure.

	3	The administrator or manager confirms his deletion.
	4	The system deletes the email form.
	5	The system deletes the email form in the database.
Extensions	**Step**	**Branching Action**

Use Case 34	The administrator or manager or User browse the reports
Goal In Context	The administrator or manager can view all the diagrams and statistics created in the page “Reports”
Scope & Level	The administrator or manager has already visited the application
Preconditions	1. The administrator or manager has successfully registered for the application. 2. The administrator or manager has entered the Page “Reports“ . 3. The administrator or manager has already created a fieldset. 4. The administrator or manager has already created a form. 5. The administrator or manager has submissions forms for a form.
Success End Condition	The administrator or manager can review his reports for the form now
Failed End Condition	Reports / System management of Reports

Primary, Secondary Actors	The administrator or manager can review his reports for the form	
Trigger	**Step**	**Action**
Description	1	The system displays a page to review the Reports.
	2	The administrator or manager views different statistics/reports for every field of the fieldset on his form.
	Step	**Branching Action**
Extensions	1	The administrator or manager has not received submission forms and the system cannot generate statistics about its form

3.5 Storyboarding

In the early stages of planning, we began using Adobe XD to develop a prototype of what our application would be. Storyboard planning was selected to help visualize our brainstorming and planning for the following key reasons:
Storyboarding helped us see the bigger picture in how we should approach the development of our project. We understood how to navigate through it. Storyboarding makes any process of evaluation and error correction trivial. Firstly, we agreed that a main menu with all the options we have described was necessary. As a result, 6 bix boxes that would help users navigate through the app were designed in our initial draft.

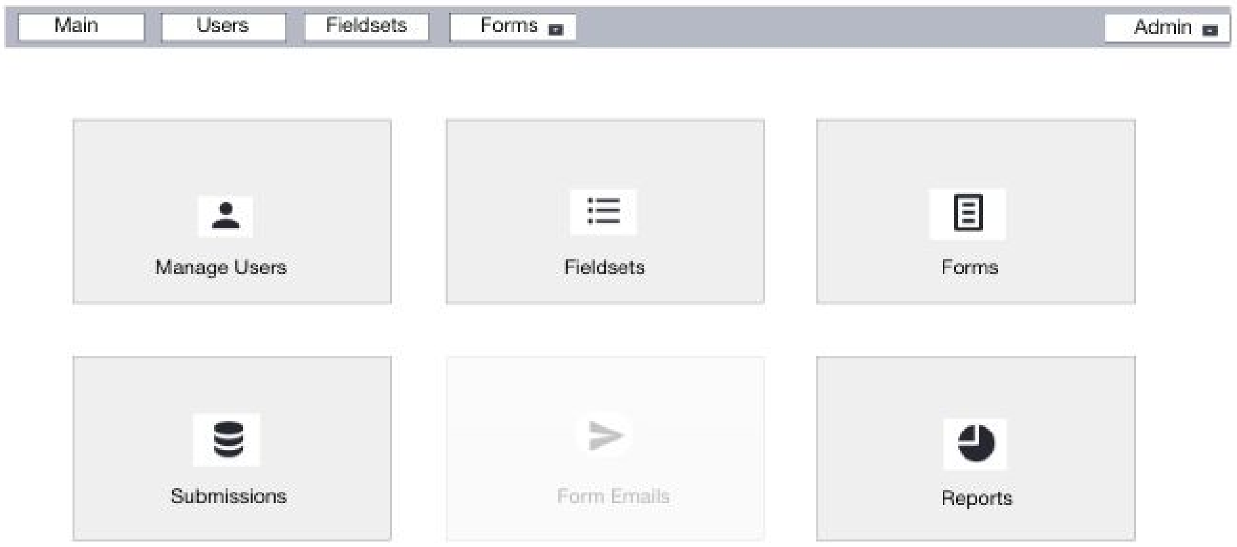
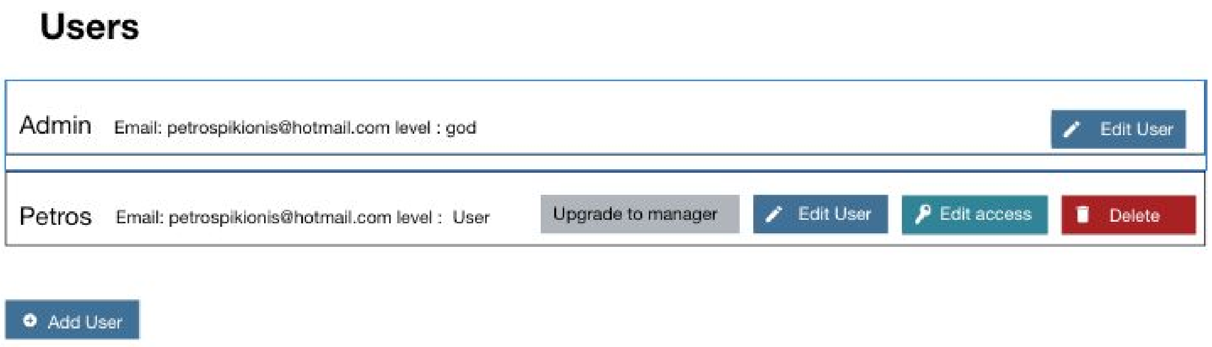


Figure 1. Dashboard

Our next goal is to construct the six menu functions. The first menu option is dedicated to the Users of the application. As a result, the administrator must have some insight on the number and identity of the application's end users. Using a list design to depict the users of the system was chosen as the more space saving and versatile option. On the left one will find the names of the users and on the right several action options are featured, such as user editting or user deletion.


Users

Admin Email: petrospikionis@hotmail.com level : god

Petros Email: petrospikionis@hotmail.com level : User

Figure 2. Users

The second menu option labelled "Fieldsets" is where all questionnaire management happens. It proves to be the most complex part of our application. Our goal is to try to visualize how a user would navigate through this Page without being overburdened by the volume of information and options given to them. This is achieved with an efficient design in addition to orderly space allocation. Once again, we decided a List was the best suited option to overcome the complexities of this Page. Upon selecting the Fieldsets menu option, a user will see a list of questionnaires along with buttons to take action on each questionnaire on the right.

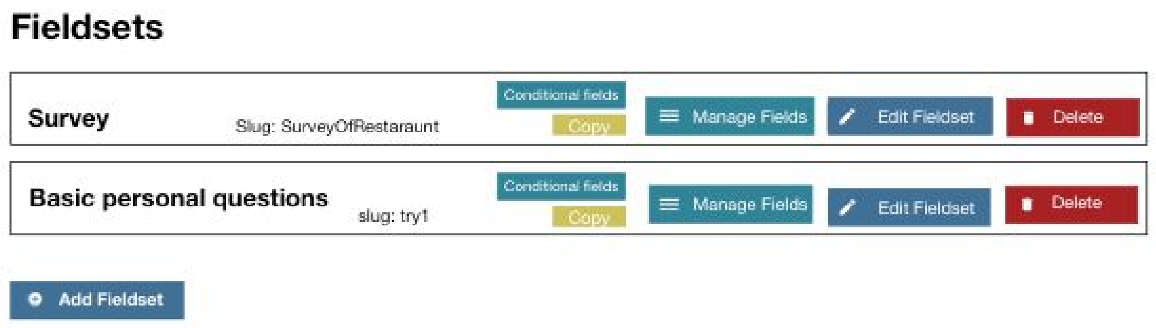


Figure 3. Fieldsets

For the next part, it quickly became apparent that we need to think small then expand. How should a question be formed in the application. A field for the actual question, and a unique question id is necessary. Furthermore, libraries within vue.js enable us to select the corresponding component that produces the type of question, for example the ‘country field’ that drops a list of countries to select from. All of this information must be aggregated efficiently. We found great inspiration from other applications that specialize in questionnaire creation, as referenced in Chapter 2.

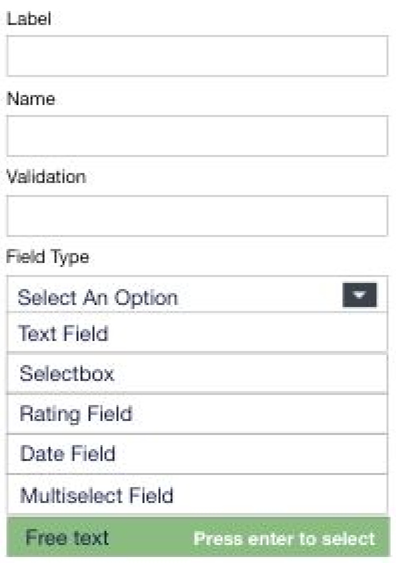
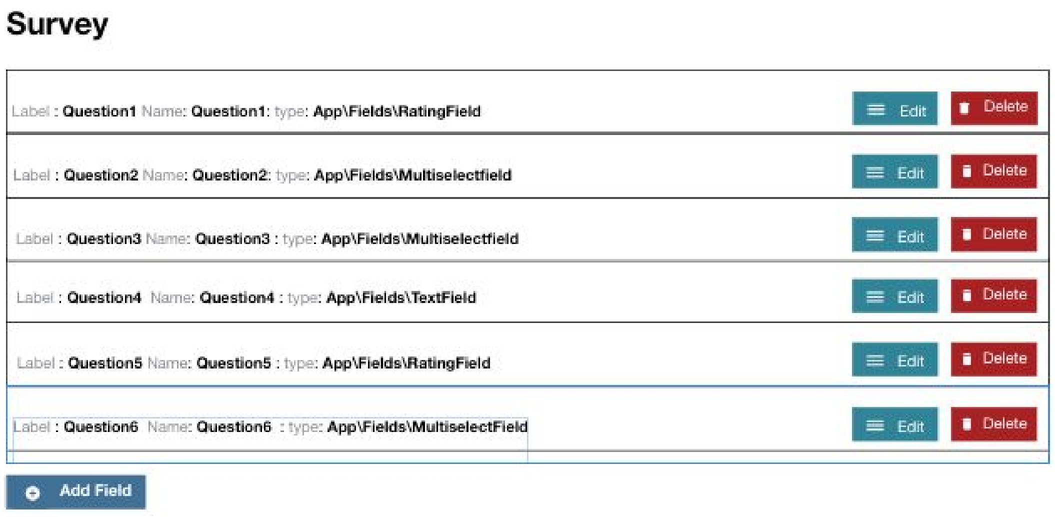


Figure 4. Field

All questions created exist in list form in each questionnaire. There are two main benefits behind this decision. First of all, during the questionnaire creation process all questions are drawn in the order they exist in the Manage Fields Page. We are also able to rearrange them through a drag and drop library. Furthermore, when using the form generator, calling questions into a questionnaire in the preferable order is a lot easier.


Survey

Label : **Question1** Name: **Question1**: type: **App\Fields\RatingField**

Label : **Question2** Name: **Question2**: type: **App\Fields\Multiselectfield**

Label : **Question3** Name: **Question3** : type: **App\Fields\Multiselectfield**

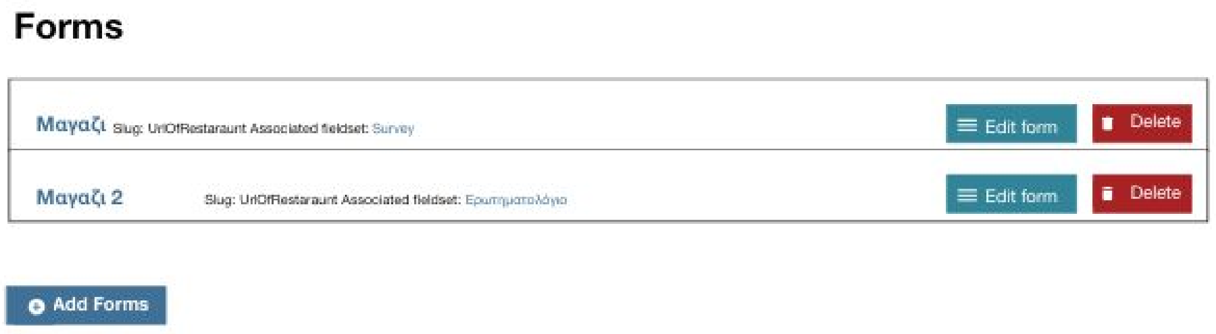
Label : **Question4** Name: **Question4** : type: **App\Fields\TextField**

Label : **Question5** Name: **Question5** : type: **App\Fields\RatingField**

Label : **Question6** Name: **Question6** : type: **App\Fields\MultiselectField**

Figure 5. Manage fields

As far as the Forms Page is concerned, a user will be presented a list of Forms, in the same format as in other Pages i.e. name of Form on the left and action buttons on the right.

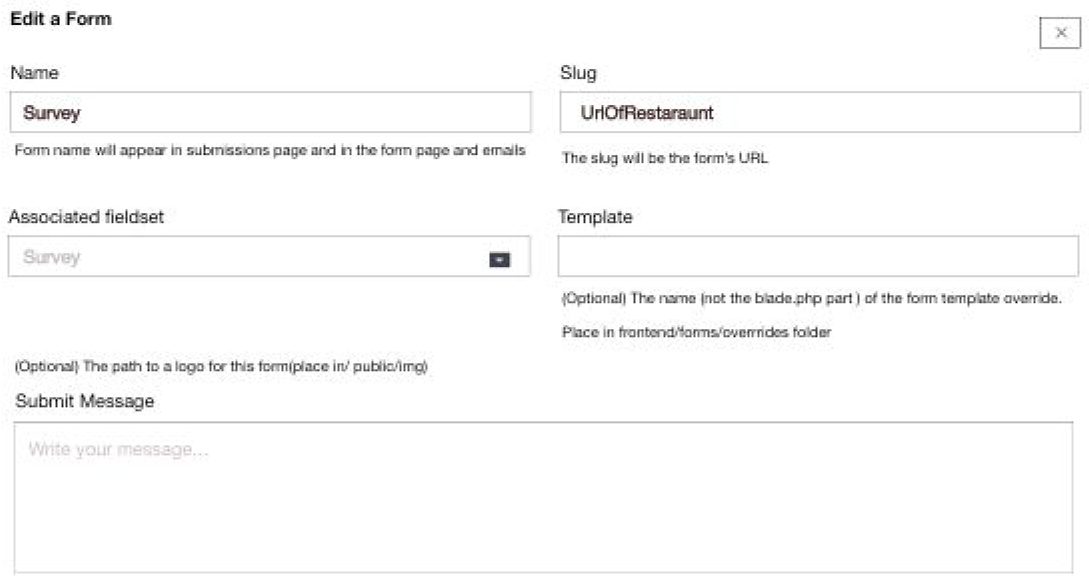

Forms

Μαγαζι Slug: UrlOfRestaraunt Associated fieldset: Survey

Μαγαζι 2 Slug: UrlOfRestaraunt Associated fieldset: Ερωτηματολόγιο

Figure 6. List of Forms

Here a user is able to fill in a form with fields such as name of questionnaire, which questionnaires it is connected to and what the presentation elements will be.


Edit a Form

Name: Survey

Form name will appear in submissions page and in the form page and emails

Slug: UrlOfRestaraunt

The slug will be the form's URL

Associated fieldset: Survey

Template:

(Optional) The name (not the blade.php part) of the form template override.

Place in frontend/forms/overrides folder

(Optional) The path to a logo for this form(place in/ public/img)

Submit Message

Write your message...

Figure 7. Form

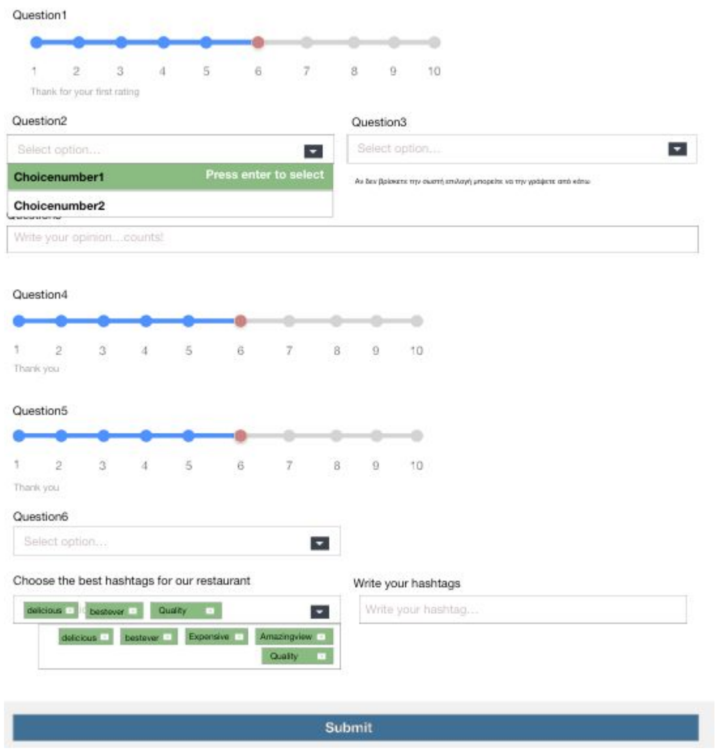
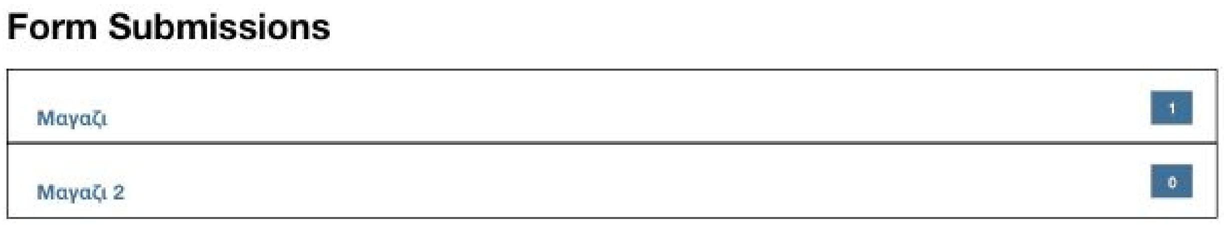


Figure 8. Survey

For this section, all answers must be structured with reference to the questionnaire they belong to. As a result, we designed it to be in List form for every questionnaire.



Form Submissions

Μαγαζι	1
Μαγαζι 2	0

Figure 9. List of submission Forms

We are directed to all answered forms for each option selected in the submissions forms list.

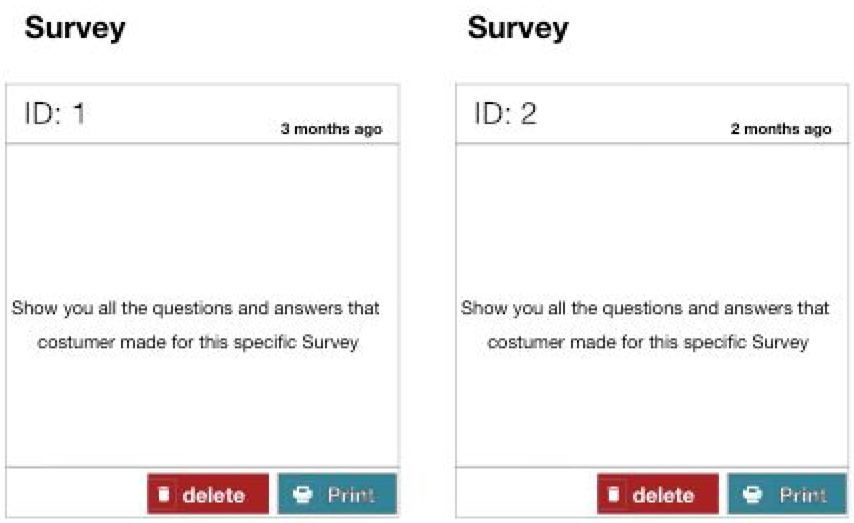

Survey
ID: 1
3 months ago
Show you all the questions and answers that costumer made for this specific Survey
delete
Print
Survey
ID: 2
2 months ago
Show you all the questions and answers that costumer made for this specific Survey
delete
Print

Figure 10. The submission Forms

The emails sections utilizes a list for every question form.

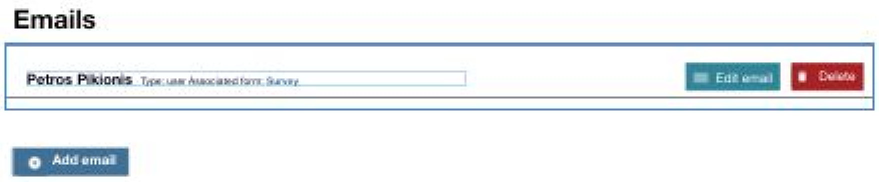
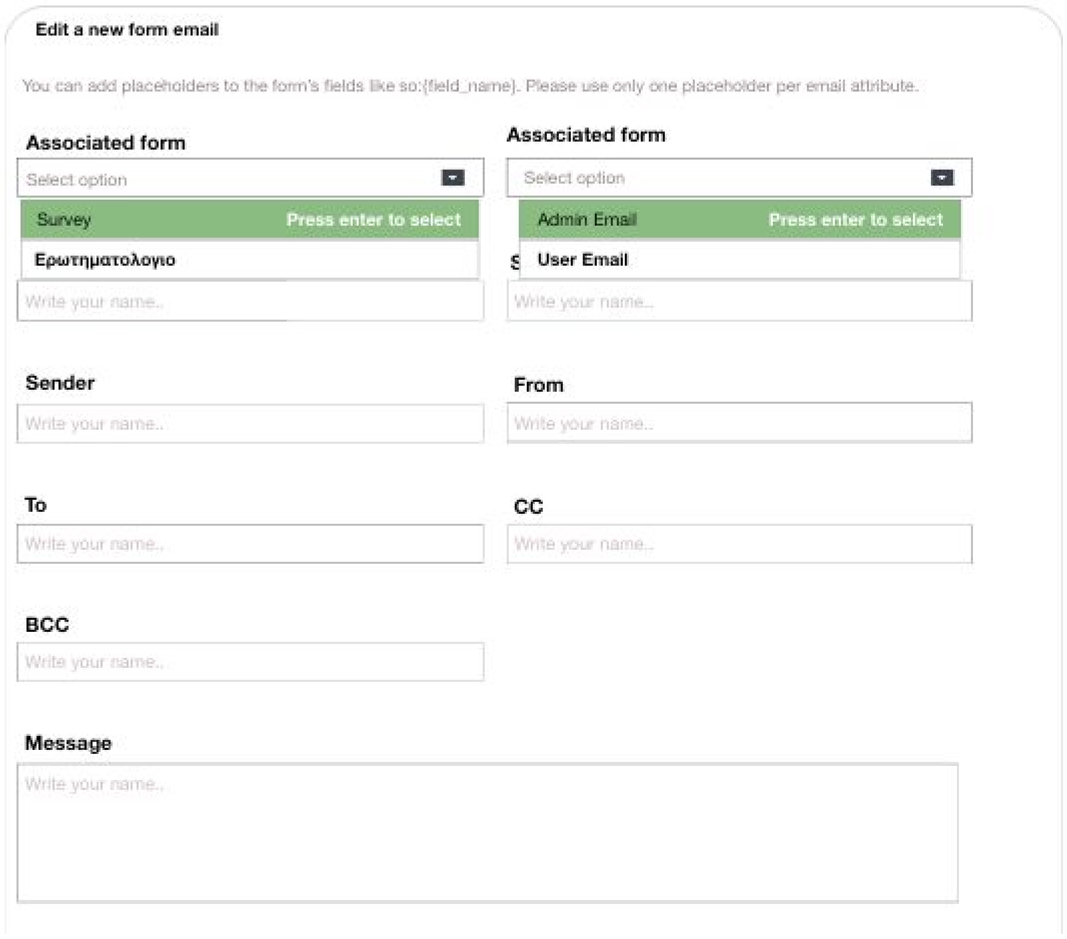


Figure 11. List of emails

Firstly, we need to know what Form it relates to, which address we're sending the mail to, what the title of the email is and then the fields that have data to send to the mail server, in order to retrieve the mails we want


Edit a new form email

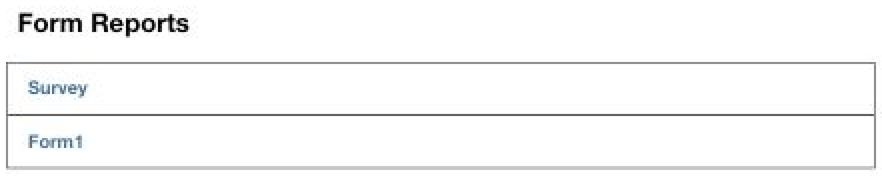
You can add placeholders to the form's fields like so:{field_name}. Please use only one placeholder per email attribute.

Associated form: Select option	Associated form: Select option
Survey — Press enter to select	Admin Email — Press enter to select
Ερωτηματολογιο	User Email
Write your name..	Write your name..
Sender: Write your name..	From: Write your name..
To: Write your name..	CC: Write your name..
BCC: Write your name..	

Message: Write your name..

Figure 12. Email Form

Lastly, we need a list of answered questionnaires to select from, whose statistical graphs will then appear.


Form Reports

Survey
Form1

Figure 13. List of Reports

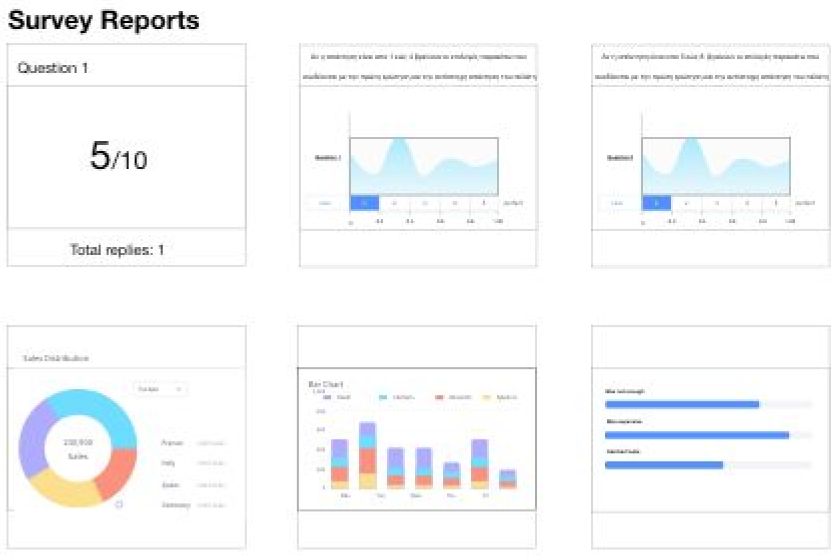


Figure 14. Reports and graphs

At the initial stages of the design process, we used storyboards to visualize and organize our ideas and communicate them to potential users in order to obtain feedback. A storyboard is a low fidelity prototype consisting of a series of screen sketches. Benefits of Storyboarding can be summarized as following: At the initial stages of the design process, we used storyboards to visualize and organize our ideas and communicate them to potential users in order to obtain feedback. Storyboards provide a bird's eye view of the system. They demonstrate the functionality of the system being designed. They demonstrate navigation across screens. They are easy to evaluate and make changes - Brainstorming provides opportunities for the creation of new ideas, and the Storyboards facilitate their presentation within the overall functionality, regardless of time and resource constraints. It is the fastest way of illustrating all functionality potential, before selecting what will be kept. The Storyboards are a first attempt at user interface design and navigation design based on the use case's description. Once storyboards have been created, it is easy to communicate with end users and refine the user requirements, and understand better the system and technology requirements for the implementation. Even though much of the functionality is fixed from the initial development process stages, some ideas are still considered for future implementation and they are taken into consideration during architectural and database design. The Administrator interface is considered the highest priority of the application. The mobile application development for the guest is considered as a low priority. These will be added in future versions.

3.6 Evaluation and Refinements

Constant assessments of progress is an important part of Software Development. Our application underwent a series of evaluations throughout the whole development process - during and after each stage of design and development - with the assistance of users selected from the community the application is being built for (as seen in the persona description above) as well as usability experts. Three users were used for the evaluation of the digital prototypes. A Cognitive Walkthrough conducted with UI experts was the first evaluation. These UI experts involved were a web designer that specializes in the development of applications in the tourism Industry, and two software developers. It produced the most valuable feedback on the user interfaces. The evaluation process involving the UI experts followed the think aloud methodology. They concluded that the User Interface was comfortable enough to use themselves, however, it would be too complicated for an everyday user to operate. A description of the systems main functions and purpose was given to the users, who were then given specific tasks to undertake. The users were encouraged to speak their minds even though no assistance was given to them. Notes on their thoughts and problems they faced were taken during this process.

3.7 Summary

In this Chapter the requirement analysis and a first draft of the project phases was described. Firstly, we created an image of the end user of our application system in our minds. Only a small number of Personas was needed to describe the requirements of our users. After giving these Personas their characteristics they were used in the analysis of use cases and the creation of a user interface prototype. A digital storyboard was created and the use cases of the system were then described.A Storyboard is a formal diagram of a potential user interface that can assist in clarifying how the end user will use the application system. It outlined the way this functionality was going to be presented in the UI as well as presenting the navigation across screens. Heuristic evaluation and think aloud methodologies were used to evaluate the digital prototypes, and the results were used to refine the interface and navigation design of the system.

Chapter 4 - Web System Architecture and Implementation Technologies

4.1 Introduction

In Chapter 4 a description of the architecture of the Web Client and the Web Server of our application is given. Due to the high expected update rate, certain features of the application, for example, the creation of the survey or the graphs- statistics analysis, are implemented via a traditional on-line architectural style, where the page preparation happens in the Web Server. Rich functionality accessible to the user by a REST service that the Web Server offers. The REST Service serves http requests achieving a desirable functionality and delivers JSON objects to the client. Our application's Web Server is structured in a MVP style of architecture. A replacement of the presentation page's parts is supported by the components, as well as dynamic graphics visualizations, and work in on-line and offline fashion. A closer look into the architecture that supports these functionalities in the Web Server and Client is given, and also, the components' functionality, as well as a description of the control logic taking place during execution. Lastly, we describe the tools and technologies that we used for this implementation.

4.2 Technologies Used

4.2.1 The basic architecture of Laravel applications

Laravel is an MVC framework for the development of modern web applications. It is a software architecture standard that separates the representation of information from users' interaction with it. The architectural standard that it has adopted has been around since the mid-1970s. It remains current, and a number of frameworks still use it today.
MVC - that stands for 'Model View Controller' - holds representation of architecture developers adopt for their application building projects. With MVC, we look at the structure of our application through the lens of how data flows in our application. A key concept in MVC architecture is the separation of application/business/domain logic for the rest of the UI, by dissecting the application in three parts: the view, the controller, and the model. The model part is responsible for the application's data management and fundamental behaviors. It follows requests, such as the retrieval of information, instructions to alter the state of its information, and even the notification of observers in event-driven systems when a change takes place. This holds true for any number of storage systems or data structures and databases. The view provides the UI layers of the application. In our case it will render our VUE components into the interface.

The controller makes calls to model objects and the view to take appropriate action, upon receiving user input. These three components connecting creates the three basic components of Model View Controller.

We have a structure that looks like this:

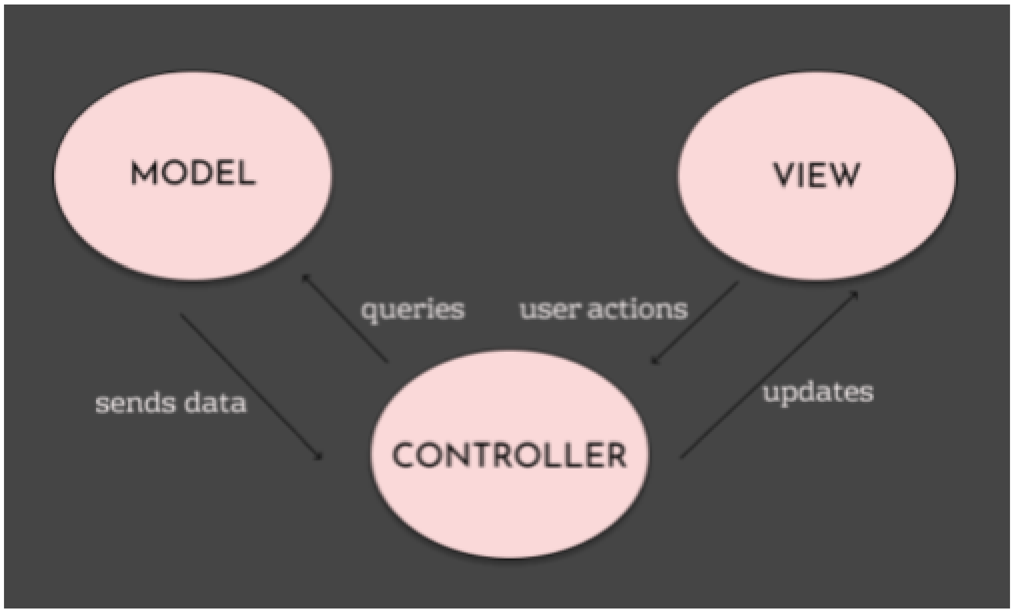


Figure 15. The MVC architecture

4.2.2 Advantages of using the laravel PHP framework

Below, some key features of PHP Laravel Framework, that make a lucrative option to developers and businesses, are presented.

Template Engine: Laravel is famous for its integrated lightweight templates, a popular tool for the creation of wonderful layouts using dynamic content seeding. Furthermore, multiple widgets using CSS and JS with robust structures are provided. Laravel templates are designed with a focus on a simplified creation of complex layout with distinctive sections.
MVC Architecture Support: Laravel supports MVC architecture which, as we described earlier, is used for the separation of presentation layers and business logic. MVC pattern of

Laravel improves the performance of the application, its security and scalability, as well as providing a lot of built-in functions.

Eloquent ORM (Object Relational Mapping): Including a simple PHP Active Record implementation, Laravel offers Eloquent ORM. Web App Developers are thus allowed to code in PHP rather than SQL for their DB queries. An ORM generally performs faster and smoother than other PHP frameworks.

Security: Substantially strong web application security is one of Laravel's main key points. Passwords avoid being stored in plain text format - a significant security hazzard - by using modern hashing and salting algorithms. In fact, a "Bcrypt Hashing Algorithm" is used for encrypted password generation. Furthermore, prepared SQL statements are used to prevent any SQL injection attacks.

Artisan: Artisan is a command line tool of Laravel framework to automate common and repetitive tasks. This automation saves development time and reduces the complexity of a Laravel project. For example, Artisan has built-in commands for user authentication, database migration file generation, and a frontend asset minifier (Saunier 2014, 73). Artisan commands are compatible with Laravel specific features. Features such as an event, middleware and policy are generated through the Artisan command line environment (Artisan console, 2019). However, Artisan commands not only generate Laravel specific features but are also extendable for project specific needs such as transferring pending mails to the recipient is obtainable through extending Artisan commands. Laravel framework provides a quality of life built-in command tool, Artisan, which automates most of the tedious programming that needs to be done. Artisan extents its utility by creating the DB structure, a skeleton code, and migration managing, ultimately, making it an easy-to-use system. Artisan can also perform basic MVC file generation through command-line. It also manages these assets and their configurations. Lastly, it encourages the creation of customized commands for developers to use as they prefer.

Libraries & Modular: Laravel is one of the few PHP frameworks that comes with pre-installed Object-Oriented and Modular libraries. For instance, an easy to implement Authentication library is included that will check active users, features Bcrypt hashing, password reset, CSRF (Cross-site Request Forgery) protection and encryption. Finally, Laravel follows modern PHP concepts, such as being divided into individual modules, which creates a well suited environment for responsive and modular web apps development.

Database Migration System: Laravel and its migration system makes expanding the web application database structure an easy and risk free task, preventing re-implementation every time there's change in code, as well as data loss. It also enables the use of PHP to perform Database changes instead of restricting developers to the use of SQL only.

4.2.3 Vue.js framework

Vue.js is a component based framework. Reusable components can be compiled as a small full functional piece of code and embedded into different pages of the application. The basic structure of a Vue component is presented below.

Vue.js is an MIT open-source framework that uses JAvascript whose purpose is interface building. Vue is primarily used in developing user interfaces (UI) and single page applications. It boasts being the most appealing lightweight front-end framework based on MVVM mode in Web applications. Below a schematic diagram of MVVM mode architecture is shown. ViewModel serves as a middleware that connects functions and data and the communication between the two.

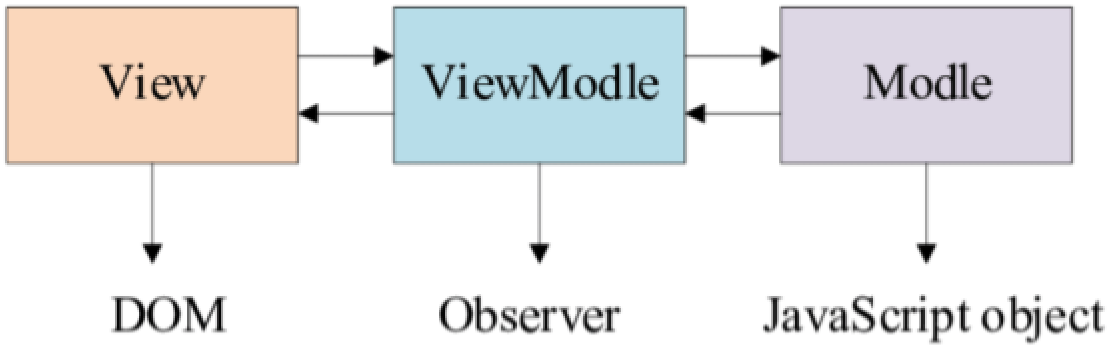


Figure 16. Schematic diagram of MVVM mode architecture

The core library of Vue.js is not a versatile framework, due to it operating only on the functional layer. It can quickly build UIs as well as implement responsive two-way data binding through a simple API. A component tree can illustrate any type of application interface through Vue.js. Repetitive development is reduced, through the use of several small components that can be reused to build a large system.

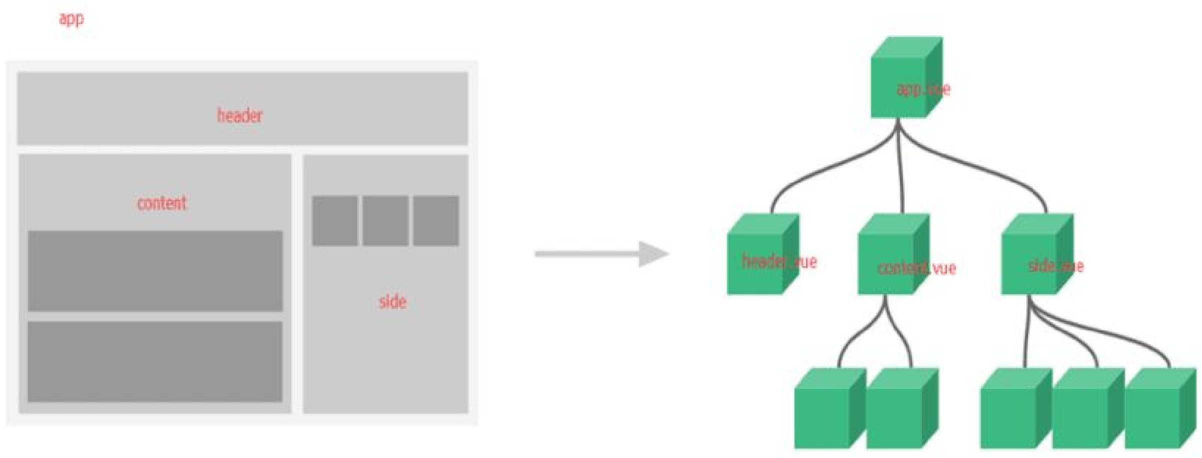


Figure 17. How a component tree can illustrate through Vue.js

4.2.4 Advantages of building applications with Vue.js

Vue.js is a great option for smaller in size projects that require submitting a form with AJAX, the value display of user-entered data , authorization and reactivity. Vue.js, a progressive framework, offers easy scalability into being optimal for larger projects as well. Router and Vuex, two of its main components make Vue.js a great solution for large single-page applications. Vue offers both application creation through the use of public APIs and also the implementation of applications that run on the server. However, Vue.js shines the most as a development tool for external API for data processing. Here are some of the best features of Vue.Js.

Very Small Size

This JS framework's success is dependent on its size. Being 18-21KB, this framework is a fast download for the user. Vue uses a Virtual Document Object Model (DOM). This creates an object representation copy of the document and allows working with a visual copy instead of the presentation itself. This way, better framework performance is achieved and our application works faster. Vue.js has an excellent performance and a deep memory allocation.

Known for its Flexibility

Vue.js allows fast writing while simultaneously running straight from the browser. It can be used to create a complex app using JSX, ES6, routing, bundling etc. This feature makes it an appealing choice for cross-platform application development. A wide variety of environments are offered. Regarding microservices and micro applications, Vue.js gives the user extensive control over the size of the application, allowing you to isolate and use only the necessary elements needed for a specific purpose. It also enables flexible migration from single-page applications to microservices by recycling parts of the previous application.

Good Detailed Documentation

A good detailed documentation is what makes developer's lives easier. Developers can write and execute their first application just following the instructions with basic knowledge of the HTML and javascript. This happens because of the good detailed documentation features.

Vue components

The easy code reusability is one of the best things that has existed with Vue.js. The entire system has built on smaller interactive views and components of the app which can easily be integrated into the existing infrastructure. That makes it easy to add small components with no negative effects on the existing template with a dedicated state manager called Vuex.

Virtual DOM of Vue.JS

The DOM (Document Object Model) API is slow. Web applications have a lot of moving parts to them that require data to be updated instantly. Sometimes your application will react differently depending on the data that was modified. Since the DOM is really slow, your web application could be unusable if the user is updating a lot of data. To solve this problem, Vue.js uses something called a Virtual DOM. Vue.js utilizes a Virtual DOM that makes rendering your application's user interface lightning fast. A Virtual DOM is a *representation* or a copy of the actual DOM and is rendered using JavaScript. In other words, when a change is made in the application, the Virtual DOM compares itself to the real DOM, defines what has changed, and *only* updates what needs to be changed.

4.3 Summary

In chapter 4 we introduced the frameworks that we used to develop our application. First, we described the MVC architectural model through the scope of the Laravel framework. A list of advantages that Laravel offers was given as well as a brief description of them. Secondly, we covered Vue.js the key points that made it an optimal option. The final paragraph is dedicated to Virtual DOM, with a thorough explanation of its operational process, offering a more fluid and dynamic experience to the end User.

Chapter 5 - Database Design Implementation and Database Interface

5.1 Introduction

In this chapter we describe our application through a Database design. The structure of the database offers category directories as well as complex associations for browsing,data mining and navigation in order to facilitate browsing and going through any information users might find desirable. Entity-Relationship diagrams are used to visualize the database design. The implementation is supported by the design of the relational schema and the database transactions.

5.2 Database Design

Users is an entity that describes everyone that may make use of the application. The application permits a User to function as an [admin, restaurant manager or waitress]. A User has a name , users_id , level, validation, password and email.

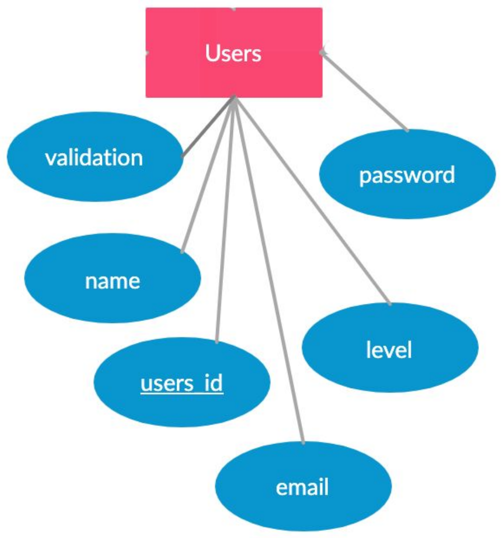


Figure 18. Users E-R diagram

Each user must have their own personal email and password to login in to the application. Each user has a personal token. A one-time password token (OTP token) is a security hardware device or software program that is capable of producing a single-use password or PIN passcode. One-time password tokens are often used as a part of two-factor and multifactor authentication. The use of one-time password tokens strengthens a traditional ID and password system by adding another, dynamic credential.

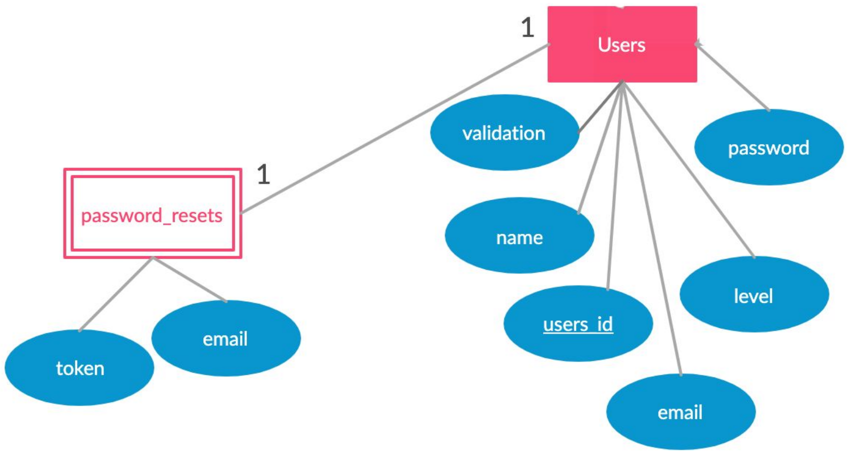


Figure 19. Users - password resets E-R diagram

The User has his own questionnaires. The questionnaires are the Fieldsets in the image below. A Fieldset's attributes are a fieldset_id, a description, a name, and a slug. The slug is an auxiliary parameter to create a URL path for the questionnaire.

The Fieldset has its own questions. The questions are the fields in the below image. A Field's attributes are a field_id, a name, a label, a note, a class, an option attribute, and a type (the type of question that will be used). For example, the type attribute may be a Checkbox field, a country field, a date field, a files field, a free text field, a multiselect field, a rating field, a repeatable text field, a select field, a text area field or a text field).

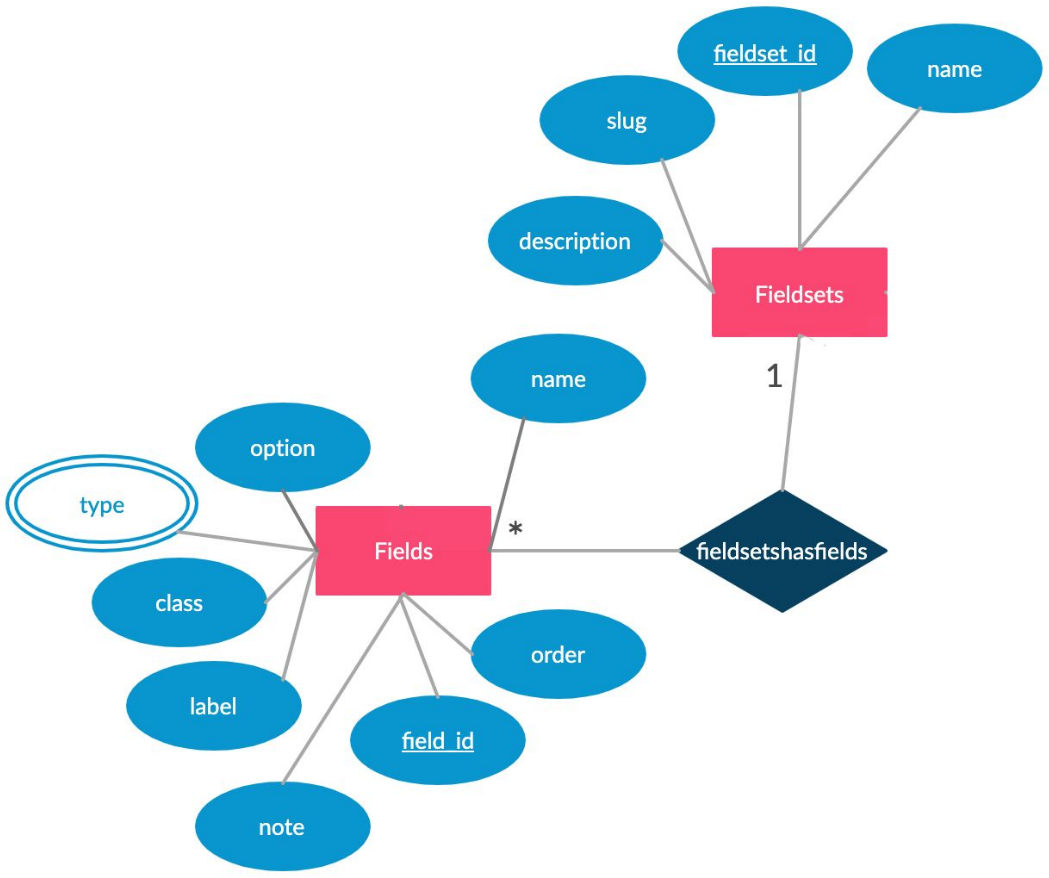


Figure 20. Fieldsets - Fields E-R diagram

The conditional fields belong to a Fieldset. We can create many conditional fields for a question. A conditional field has a conditional_field_id, a field_id, a comparison_type, a comparison, a value, a condition, an order and a fieldset_id. The “options” attribute refers to the selection of a specific Fieldset.

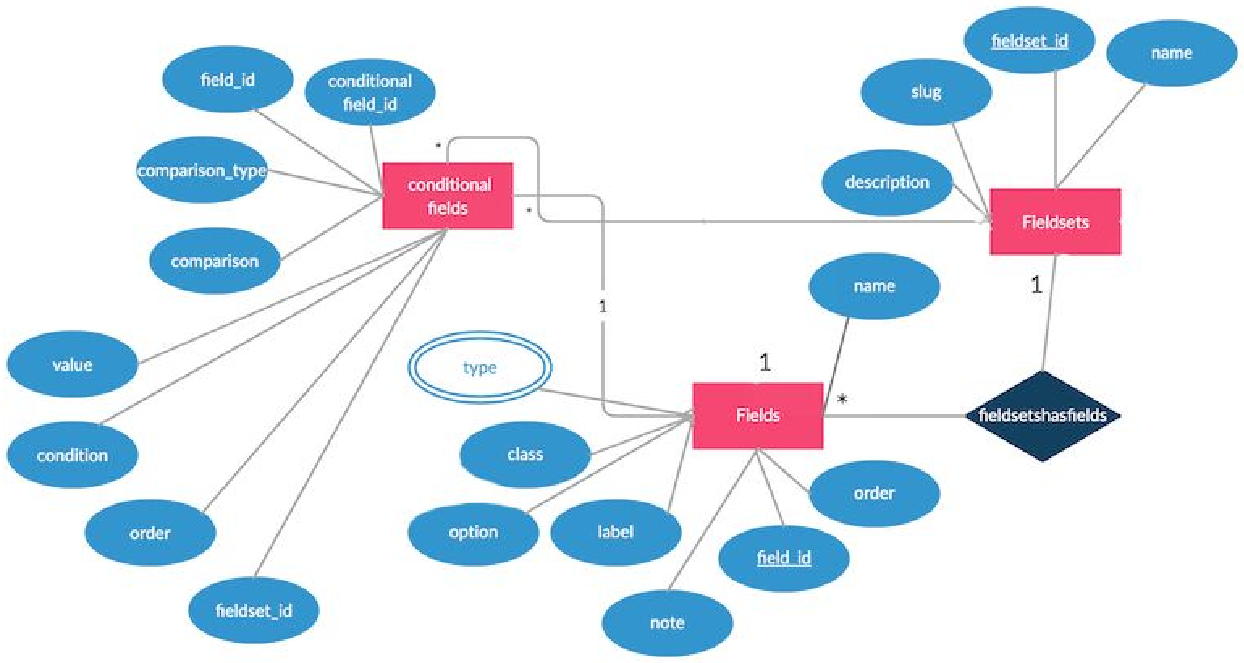


Figure 21. Conditional fields - Fields - Fieldsets E-R diagram

The Fieldset belongs to a Form, which belongs to a business owner. The business can create many Fieldsets. A form has a form_id, a name, a slug, an option and the fieldset_id. The "options" attribute refers to the selection of a specific Fieldset.

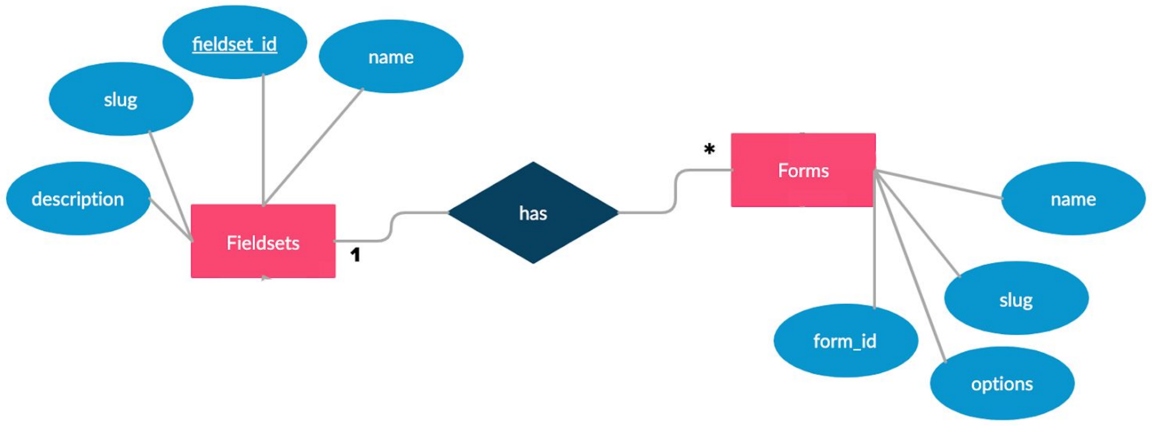


Figure 22. Fieldsets - Forms E-R diagram

The Form is also associated with an Entity called form_submissions. The form_submissions are the customer-answered Fieldsets, which belong to a specific Form. The Form_submission's attributes are the client_ip, the form_submission_id and the form_id.

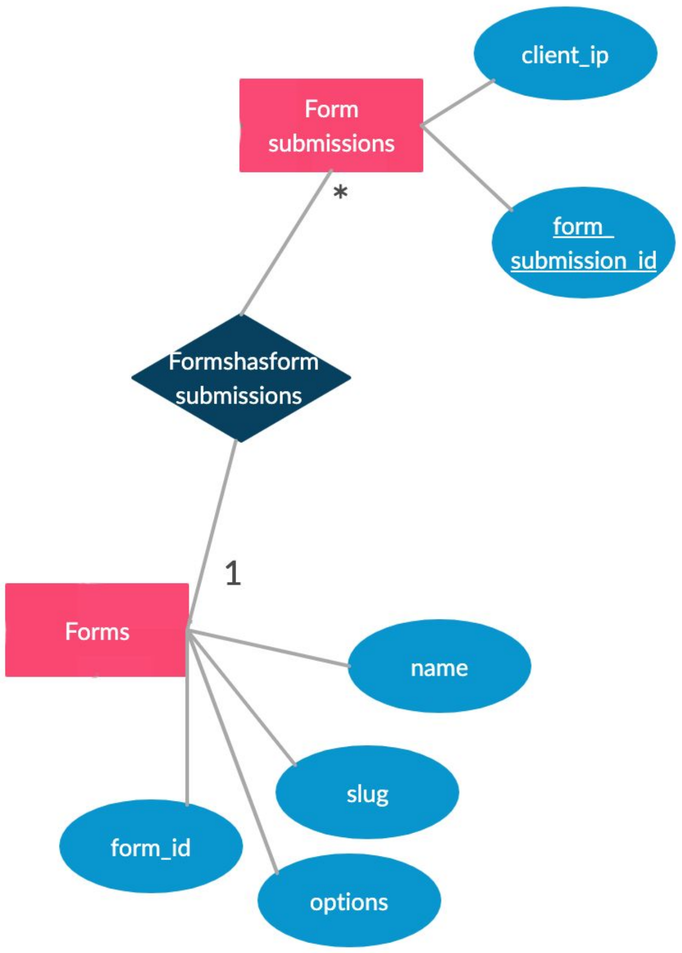


Figure 23. Form submissions - Forms E-R diagram

The Form_submissions has form_replies. The form_replies are the given answers by the customers which belong to a specific form. The Form_submission has as attributes the client_ip and the form_submission_id.

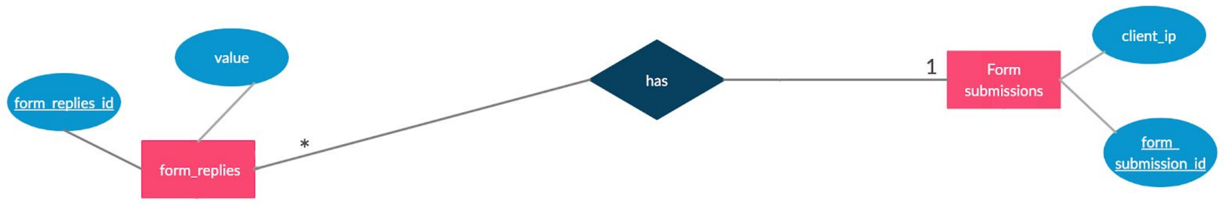


Figure 24. Form_replies - Form submissions E-r diagram

The Fields have form_replies. The form_replies the answers given by customers to the field.

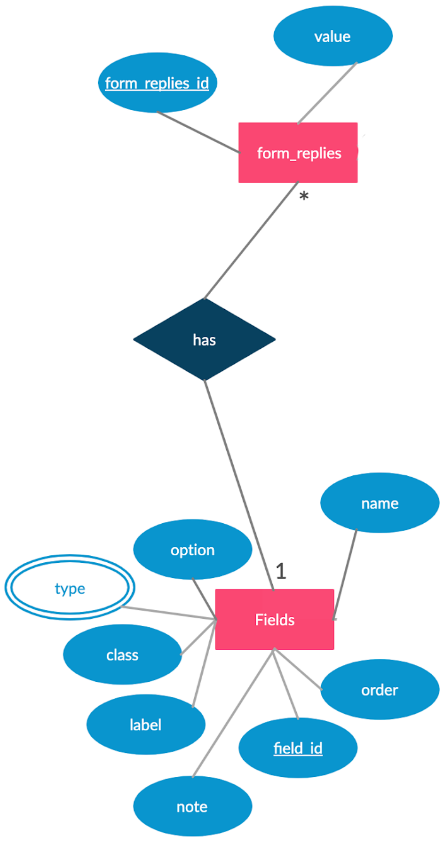


Figure 25. Forms_replies - field E-R diagram

The form_replies belongs to a Form. We must associate each answer (reply) with a Form.

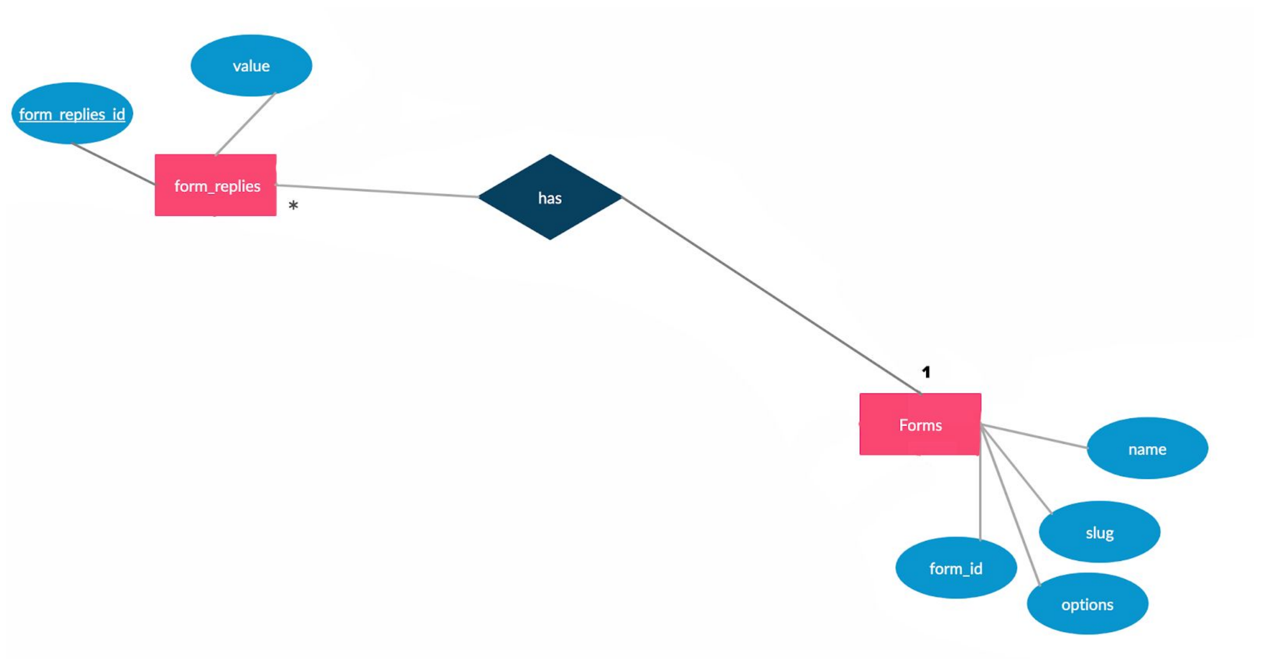


Figure 26. Form_replies - Forms E-R diagram

The form_replies has a form report. We will create statistics and graphs based on the many answers given to a question.

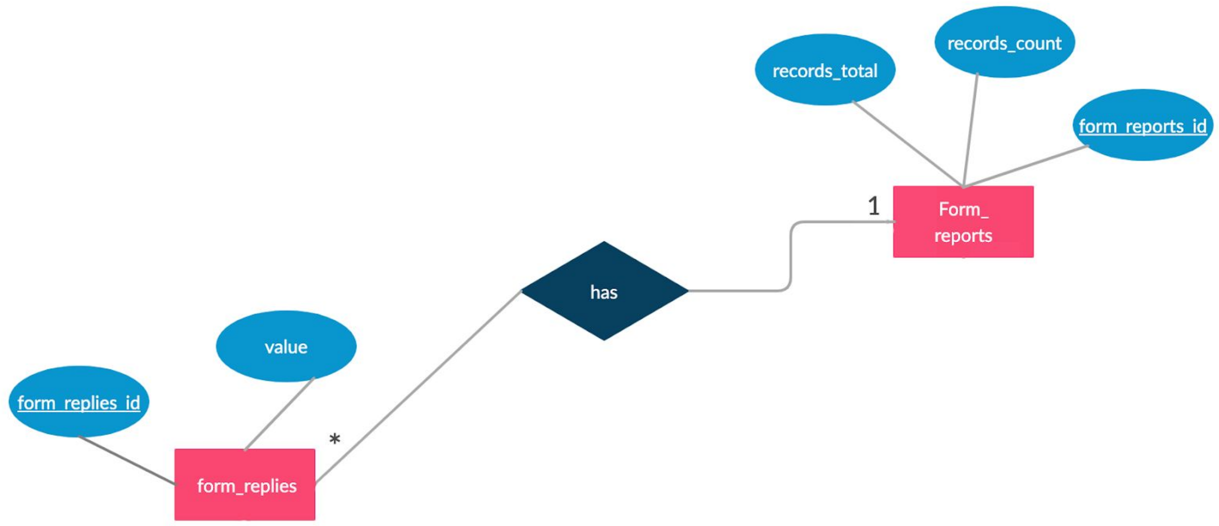


Figure 27. Form_reports - Form replies E-R diagram

The Form has Form_reports. The Form_reports creates the statistics and the graphs using the answered submissions forms. The Form_reports Entity has three attributes in total:
The record_total, the records_count and the form_reports_id.

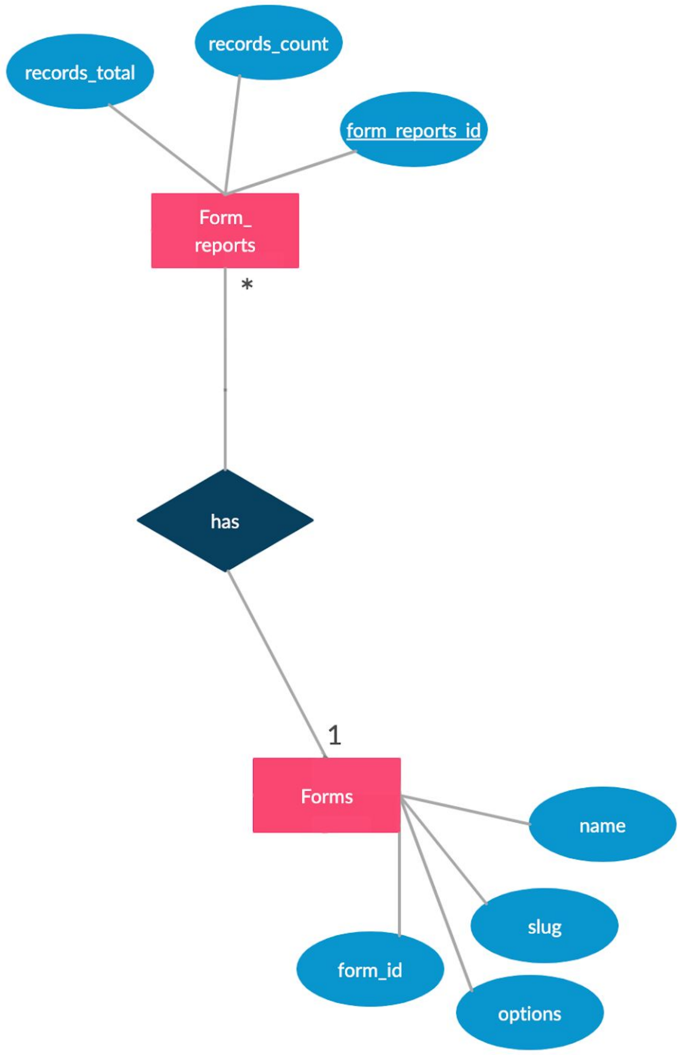


Figure 28. Form reports - Forms E-R diagram

Finally, the User has Form_emails. The attributes associated with the Foms_emails Entity are the form_emails_id , type, sender, from, to, cc, options, template, bcc, name, subject.

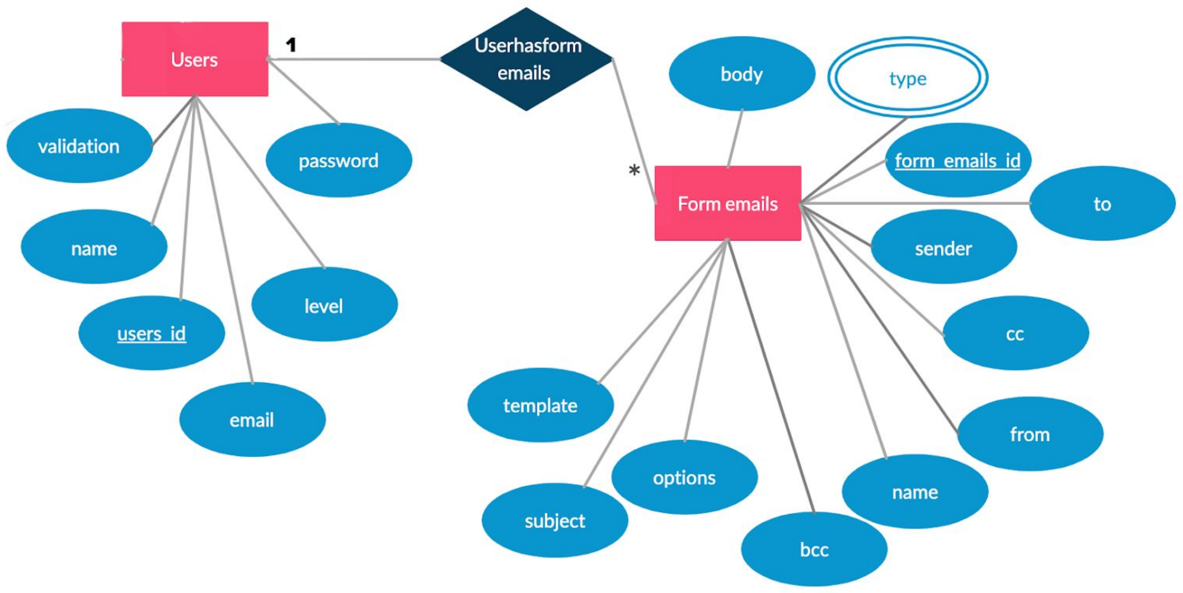


Figure 29. Users - Form emails E-R diagram

The complete Database Entity Relationship Schema is shown below:

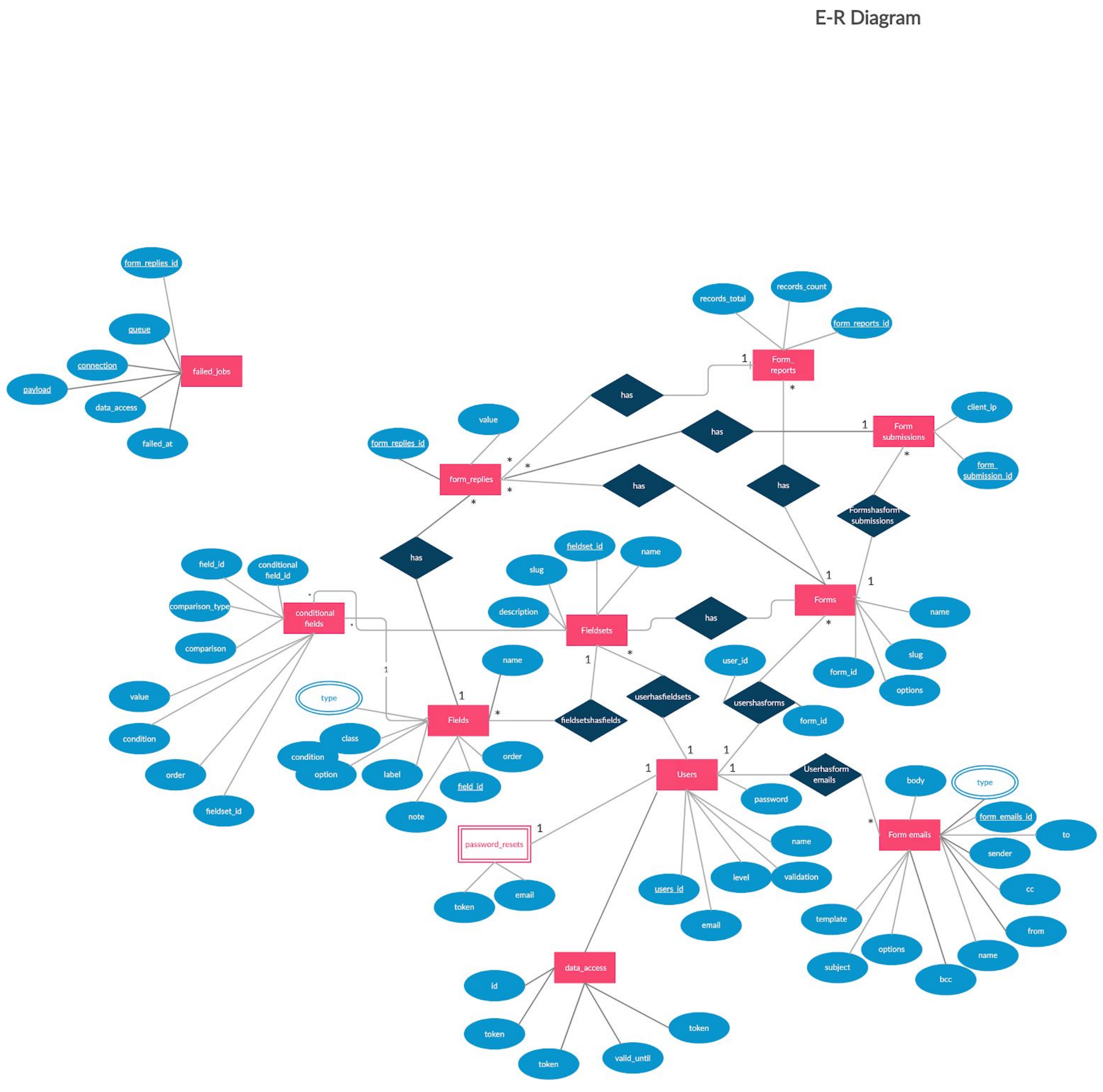


Figure 30. Database Entity Relational Schema

5.3 Relational Schema

The ER schema of the database is translated into a Relational Schema that reflects the tables used in the database. The Relational Schema consists of two Column lists, the Table Name and the Attributes. Each Table Name item is given with its associated Attributes to the right of it. Attributes that function as primary keys are underlined, ex: this_is_a_key, and all foreign keys that are used to keep information about a relationship are shown with (FK) beside them, ex: this_is_a_foreign_key (FK). Users Entity is translated into a table with no extra attributes that function as foreign keys. Form_user is an assistant Entity which combines the user_id and the form_id as foreign keys. Fieldsets Entity is translated into a table with no extra attributes that function as foreign keys. Fields Entity is translated into a table with a foreign key to the fieldset it belongs to. Forms Entity is translated into a table with a foreign key to the fieldset it belongs to. Form_emails Entity is translated into a table with a foreign key to show which form is associated with it. Form_submission entity is translated into a table with a foreign to the fieldset it belongs to. Data_access entity is translated into a table with three foreign keys to the fieldset, the field and the form it belongs to. Form_replies entity is translated into a table with three foreign keys to the fieldset, the field and the form it belongs to.

Table Name	**Attributes**
Users	id, name, email, password, level, remember_token(FK)
Fieldsets	id, name, slug, description
Fields	id, label, name, type, class, note, validation, options, order, fieldset_id(FK)
Form_emails	id, name, type, subject, sender, from, to, cc, bcc, template, body, options, form_id(FK)
Form_replies	id, value, form_id(FK), formsubmission_id(FK), field_id(FK)
Forms	id, name, options, fieldset_id(FK)
Form_submission	id, client_ip, form_id(FK)
data_accesses	id, email, token, client_ip
failed_jobs	id, connection, queue, payload, exception

form_reports	id, field_id, field_id(FK),form_id(FK),option,value, records_count,record_total,
Conditional_fields	id,field_id(FK), comparison_type, comparison, value, condition, order, fieldset_id(FK)

Relational Schema of the database, description of all the relations and their attributes. Primary keys are underlined and foreign keys have an (FK) to their right.

5.4 Summary

In this Chapter we illustrated and talked about the design of our Application's Database. The Database supports structuring to allow complex points of view and navigations for information mining and information association. We described the Entity Relationship Model, and the Relational Schema that implements the database.

Chapter 6: Architecture and Implementation of the web application

6.1 Introduction

In this chapter we describe the design architecture of our application, we analyse its components and provide an in depth analysis of its operational methods. Furthermore, we bring certain parts of our system architecture into focus. The development of our application follows the principles or Rich Internet Application (RIA), minimising the transferring of data between server and user, enabling a dynamic user - UI interaction. As a result, any user request will trigger a smarter handling of information and not a complete refresh of the page. The end product offers a user-friendlier and more intuitive experience to the end user. The existence of well established and reputable design archetypes accelerates the development process, and, in addition, they provide sustainable solutions to the more common issues that arise in software design. The server-side part of our application was developed making use of the multi-tier architecture, whereas the user-side part was implemented using the Model View Controller (MVC) pattern.

The server side: The architecture-related part of the server follows multi-level design standards and consists of Business Logic Layer,Data Layer and the Service Layer. This ensures easy maintenance of the system (system’s maintainability), reuse of its various components (reusability of the components), scalability, robustness and security.

Service Layer: The Service Layer is the controller between the parts of the client and server application, providing all the required services to the customer's components. These services have been developed as RESTful and are the middle-ware part of the application that hides business logic from the customer. They are responsible for creating, retrieving, updating and deleting users, questions, questionnaires and forms.

Data level: The data level is the part of the architecture that provides simplified access to the data stored in the system database. Specifically, it is responsible for connecting to the database and storing, retrieving, processing and deleting data.

Customer side: This part of the application is about user interaction. All actions performed by someone who manages the system are dealt with by the logic of the customer side, which undertakes the communication with the server as well as the presentation of the information. Distinction of customer side data is achieved through the Model View Controller (MVC) design model. MVC model ensures the separation of the tasks of visual presentation and event handling and their assignment to the distinct entities of the Promotion and the Auditor respectively. Some of the advantages of this approach are: (a) expanding the code that can be tested automatically (websites that contain HTML elements are difficult to test), (b) sharing program sections between pages that require the same behavior, and c) the separation of business logic from user interface logic, which makes it easier to understand, correct and maintain code. The details of the client's side architecture are described in more detail below.

6.2 The Root Directory

The web root is the publicly-accessible base folder in our web application.

The App directory: The core code of our application is enchanted in the app directory, the classes are in this directory.

The Console directory: All of the custom Artisan commands are on the console directory. The directory contains the console kernel too. We will find the custom Artisan commands which have been registered and the scheduled tasks.

The Events directory: 2 main events exist in that directory. The first one is when a form submission is deleted and the second when a form submission is received. An event class is a data container which holds the information related to the event.

The Exception directory: The Exception directory contains your application's exception handler and is also a good place to place any exceptions thrown by your application. If we would like to customize how our exceptions are logged or rendered, we should modify the Handler class in this directory.

The fields directory: The fields directory contains all the types of questions we will be able to choose. I created my own form builder which will contain only the types of questions. I chose to do it this way as installing small vue components for each field rather than a big ready-made code would be costly, useless and complicated in the development of my application.

The Helpers directory: The Helpers directory we made to help us. Contains a file in which we have 3 functions. Those functions help us to convert an associative array to an object JSON for the vue select component, to get a value inside the field and to replace a field with another field.

The http directory: This directory contains the form requests, middleware and the controllers. All the logic of our application exists in this particular one directory.

The controllers directory: The Controllers directory handles the request logic within the single class. Laravel framework follows the MVC (Model View Controller) architecture in which controllers act as moving the traffic back and forth between model and views. The controller classes will define and organize all of your route-level logic in a file called routes.php. Laravel sets up a bunch of methods to help us with the productivity inside of controllers as index, create, store, show, edit, update and destroy.

The authentication directory: LoginController contains the logic to authenticate existing users and store new users in the database. The other controllers contain actions such as displaying, storing, upgrading, deleting or resetting the questions,the questionnaires, the forms, the emails, the replies and the reports.
The Middleware directory: The Middleware directory contains some files. HTTP Middlewares are useful because provide a mechanism to filtering the HTTP requests which enters in our application. Laravel checks for the authentication by this HTTP request to verify each user. Middleware is a bridge between the request and the response. It is a type of filtering mechanism.
The authentication file: The application can verify a user through the existence of that middleware on Laravel framework. If the user is authenticated, the middleware will proceed the request further into the application.
The Cookies & Encryption file: Laravel can encrypt all cookies generated by Laravel so that they can't be modified or read by the client.
The Configuring Trusted Proxies file: We can quickly customize the load balancers or proxies in our application by TrustProxies. When our application is running sometimes it doesn't generate HTTPS links but now we can balance the traffic on port 80 with this feature of Laravel.
The TrimStrings file: The Trim Strings middleware will automatically trim all the request data.
The VerifyCrsfToken file: We protect our application by a Cross Site Request Forgery with CSRF tokens. CSRF tokens are strings that are automatically generated and can be placed in our form when we create it. The verifyCSRFToken middleware handles unique POST requests from each one form.
The requests directory: The requests directory determines if the user is authorized to make requests and to get the validation rules that apply to the inputs in the field, the conditional field, the email form and the submission form.
The jobs directory: Our application handles all the queueable jobs by the Jobs directory. Jobs run synchronously with the current request lifecycle.
The Listeners directory: Our events are handled by the classes which are on the Listeners directory. When it receives an event instance the perform logic is responsible for firing the event. We have created 3 listeners, the first and second is to generate the form for the reports or regenerate it when new data comes in our database. The last listener is to send an email after our customer submits his answers.
The Mails directory: This directory contains the classes for representing emails sent by our application. Mail object is responsible to encapsulate all the logic for building an email in a simple, single class.
The Policies directory: This directory is responsible for the authorization policy of our classes. Policies are used to determine if a user can perform a given action against a resource. If the level of the user is “ god “ or “ manager “, they have the authorization to enter almost all sections of the application.

The Providers directory: This directory is responsible for providing services in our application. Its binding services in the service container, registering events and performing other tasks for the upcoming requests in our application.
The Traits directory: The Traits directory has the file “ChangeOrder” which handles reordering a field based on an "order" column in its table. It expects that the order is already correctly set, so it just gets all the fields, removes from the Collection the one that has the changed order and reorders everything else. Then it sets the new order for the moved item.
Models: MVC framework, the letter “M” stands for Model. The business logic is handled by the Model in any MVC framework based application. Model is a class and represents the logic of our structure and the relations between our underlying data tables. Each of the database tables has a corresponding “Model” and allows us to interact with that table. With Models we can retrieve, insert, and update information into our data table. Models are stored in the main app directory of Laravel.
The bootstrap directory: This directory has the app.php file which bootstraps the framework. This directory contains the cache directory and generates files for performance optimization such as the route and services cache files.
The Config directory: The config directory, as the name implies, contains all of our application's configuration files.At a high level, configuration files are located in the config/ folder of a Laravel project. The framework ships with various configuration files that make it easy for you to do things like pick which database driver you want to use and define external services.
The database directory: This directory is responsible for our model factories, database migrations and seeds.
The factories directory: This directory is responsible for allowing us to build fake data for our models. It is very useful for testing and seeding fake data into your database to see your code in action before any real user data comes in.
The migrations directory: Laravel migration is a way that allows you to create a table in your database. A migration is an easy way to avoid writing your table structure bothering with SQL queries. Migrations allows you to make incremental changes, roll back changes, and also keep the database structure in sync. Eloquent gives us an easy way to control our database to do things like accessing the records, deleting records and more. Eloquent makes tasks like adding, deleting, and updating the database easy and without the need of writing complex queries.
The public directory: This directory is responsible for the index.php file, where all the requests start entering in our application and configures autoloading. This directory contains assets such as images, JavaScript, and CSS.

The resources directory: This directory is responsible for the views as well as your raw, un-compiled assets such as LESS, SASS, or JavaScript. This directory also houses all of your language files.

The assets directory: This directory is responsible for all our vue components . Each model has its own vue components. When you compose your application with Vue components, each component's dependencies are automatically tracked during its render, so the system knows which component actually needs to be updated when the data are changed. This makes improving the overall application efficiency by updating DOM with minimal resources. In Chapter 8 we will see how our vue components will appear to our customers.
The classes directory: The classes directory has 2 files. The first is for the errors, what we have made is τ ο determine if an error exists for a given field, retrieving the error message for a field, recording the new errors and clearing the errors.
The components directory: The components directory contains all the vue components , Each module (Users, fieldsets, fields, forms etc) have their own vue components. With Vue Components we create custom elements, which can be reused in HTML. Components are independent units of an interface. They can have their own state, markup and style.
The helpers directory: The Helpers directory contains a file with all the vue components which are associated with all the types of question. What we need to point out is that for each type of question, we have created a vue component which every time we choose this type of question our system brings this component and adds it to our questionnaire.
The views directory: The view directory contains the HTML served by your application, and serves as a convenient method of separating your controller and domain logic from your presentation logic. I have split my files per section.
The routes directory: This directory is responsible for the route definitions in our application. Route files are included web.php, api.php, console.php and channels.php. The api.php file is responsible for routes as the RouteServiceProvider places in the api middleware group, which provides rate limiting. These routes are intended to be stateless, so requests entering the application through these routes are intended to be authenticated via tokens and will not have access to session state. The console.php file is responsible for defining all of Closure based console commands. Each Closure is connected to a command instance allowing a simple approach to interacting with each command's IO methods. Even though this file does not define HTTP routes, it defines console based entry points (routes) into your application. The channels.php file is where you may register all of the event broadcasting channels that your application supports. The web.php file is responsible for the RouteServiceProvider places in the web middleware group, which provides session state, CSRF protection, and cookie encryption. All of our routes will be defined in the web.php file.
The storage directory: This directory is responsible for compiling Blade templates, file based sessions, file caches, and other files generated by Laravel. This directory is segregated into app, framework, and logs directories. The app directory may be used to store any files generated by your application. The Laravel directory is used to store generated files and caches. Finally, the logs directory contains your application's log files. The storage/app/public directory is hosting profile avatars that should be publicly accessible. You should create a symbolic link at

public/storage which points to this directory. You may create the link using the php artisan storage:link command.
The tests directory: The tests directory contains your automated tests. Each test class should be suffixed with the word Test. Laravel is built with testing in mind. In fact, support for testing with PHPUnit is included out of the box and a phpunit.xml file is already set up for our application. The framework also ships with convenient helper methods that allow you to expressively test our application. Our application's tests directory contains two directories: Feature and Unit. Unit tests are tests that focus on a very small, isolated portion of our code. In fact, most unit tests probably focus on a single method. Feature tests may test a larger portion of our code, including how several objects interact with each other or even a full HTTP request to a JSON endpoint.
The vendor directory: The vendor directory is responsible for our Composer dependencies.The file .env has *all* of our application's **environment variables**. Composer is a PHP application that can manage PHP libraries, tools and frameworks that we use in our application. Composer.json describes the dependencies and contains other metadata as well. The composer.lock file is an *exact* record of the dependency versions that have been installed.

6.3 Summary

In Chapter 6 we delve into the different directories of files that we structure our application. Each Paragraph dedicates itself to the different directories that implement the end product, along with a detailed description of the files that take place in them. The separate analysis of each directory denotes a unique functional purpose for each one of them. In addition, we mention how some of these directories are automatically generated, while others are specifically created.

Chapter 7 - User Interfaces

In this chapter we describe the basic principles we followed in designing the User Interface. Section 7.2 refers to the page where the user connects to the application. Section 7.3 refers to the page where users are located and their responsibilities. Section 7.4 refers to the page where users can create questionnaires. Section 7.5 refers to the page where users can create the image of their store. Section 7.6 refers to the page where users can see the answers to the questionnaires given to their customers. Section 7.7 refers to the page where users can find the statistics and graphs automatically generated by the imported, filled in questionnaires.

7.1 Login Page

The figure provided below shows the login form of the system. The login form includes a field for the username and the password of the user, as well as a Remember Me option. It also displays an appropriate error message when the username given or the password fail to confirm. Access to the application can only be granted by the administrator.

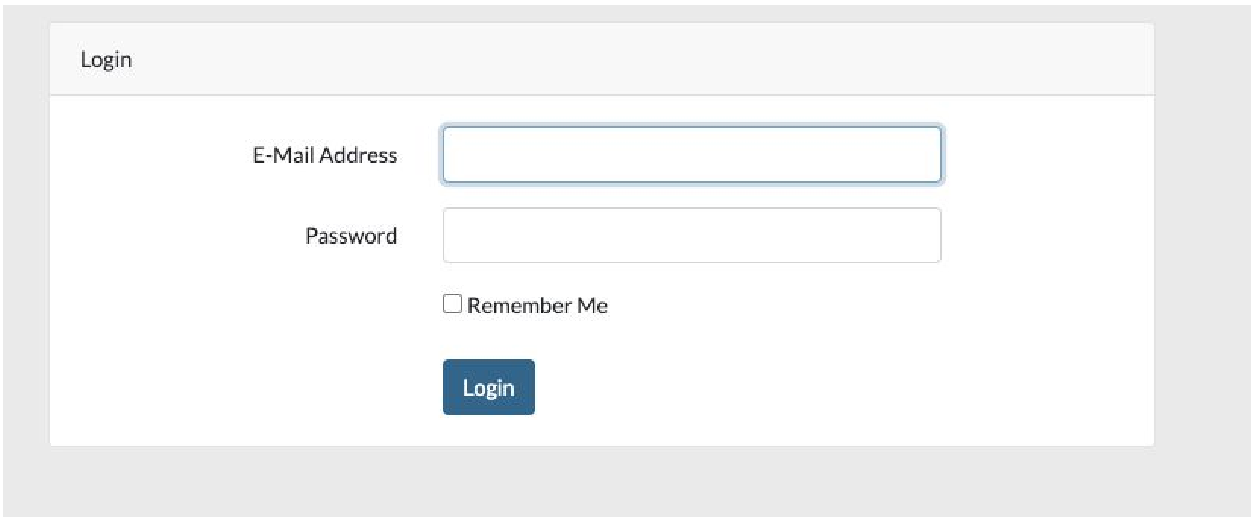


Figure 31. login Page

7.2 Home page

The figure below refers to the home page of the application. The user will be introduced to a panel with six different options as well as a header menu.In the header menu the user will be able to select between the Main button that takes him to the home screen, a Users, a Fieldsets button, and finally a Droplist option that takes you to the Forms, Submissions, Form Emails and Reports pages. The header menu is retained in all the application pages. The active screen options of the home page are Users, Fieldsets, Forms, Submission Forms, Email Forms and Form Reports.

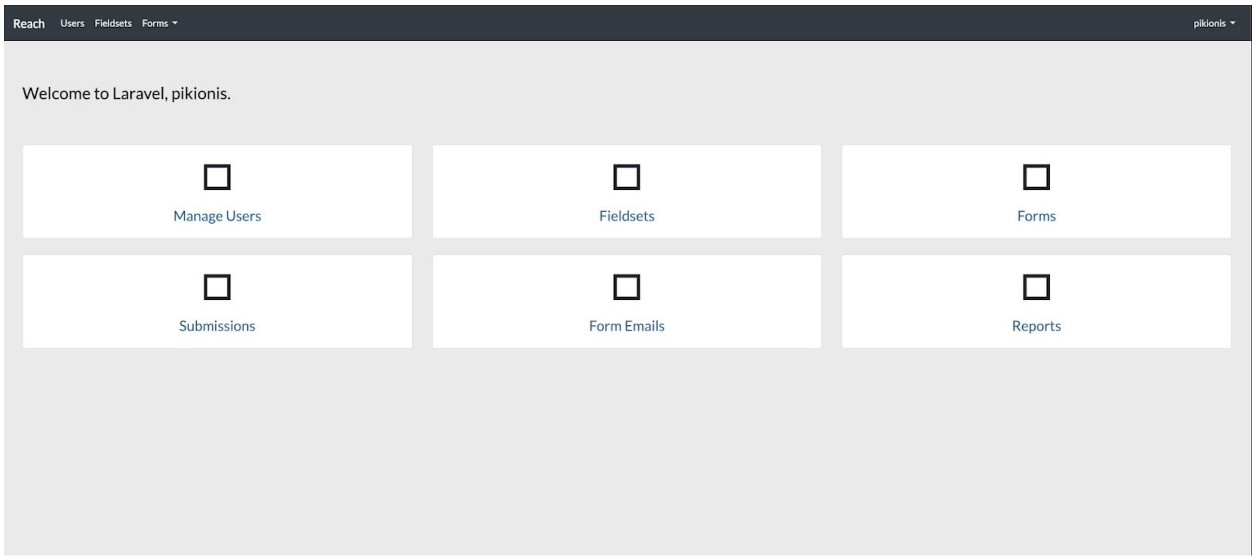
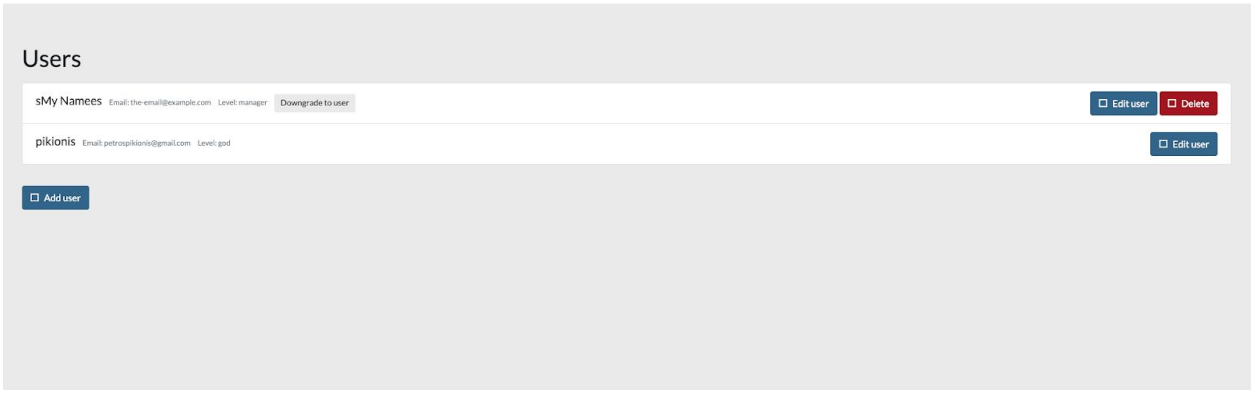


Figure 32. Main Dashboard

7.3 Users page

The Users page shows all the users that have been added into the system. The administrator is able to add any user, i.e. a business owner or a staff member. The Users page has a system of privileges given to the users registered. A manager is able to add their staff as Users, however, they have limited access to the applications resources.



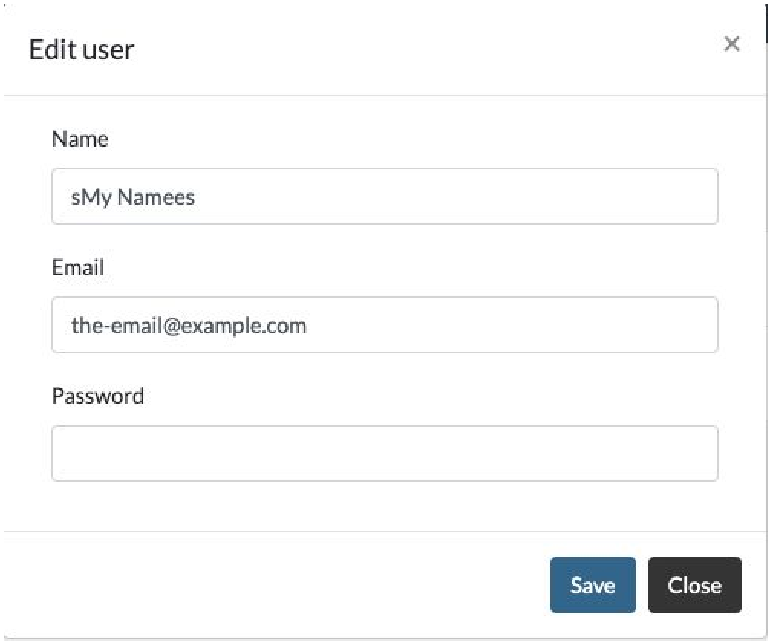
Users

sMy Namees Email: the-email@example.com Level: manager

pikionis Email: petrospikionis@gmail.com Level: god

Figure 33. Users Page

This section gives the options to add, to edit or remove a user. This section is designed to give accessibility to the users to view the questionnaires, the customers responses and the reports.



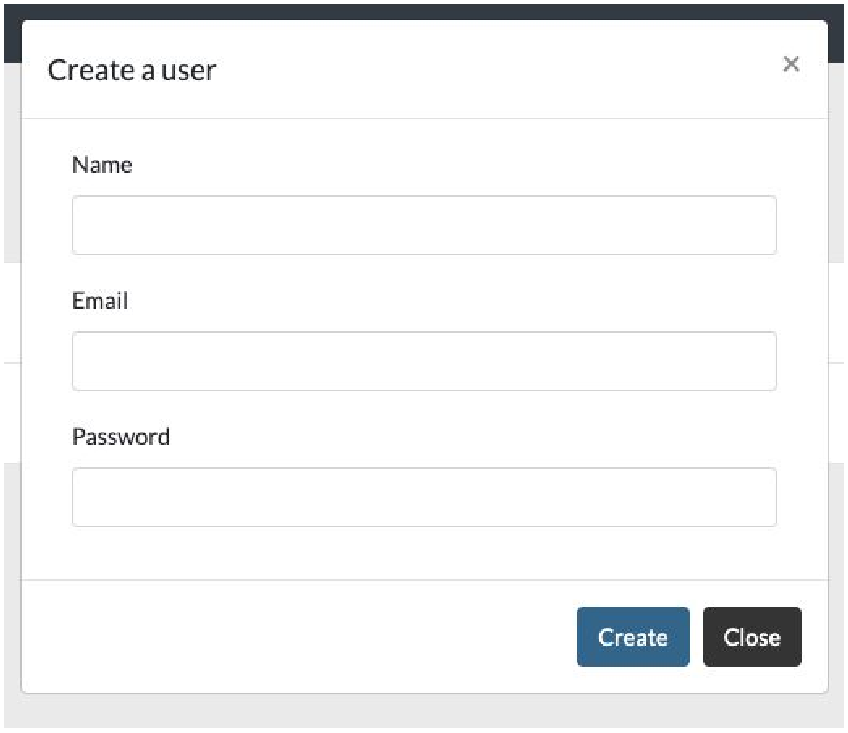
Edit user

Name: sMy Namees

Email: the-email@example.com

Password:

Figure 34. Edit a user


Create a user

Name

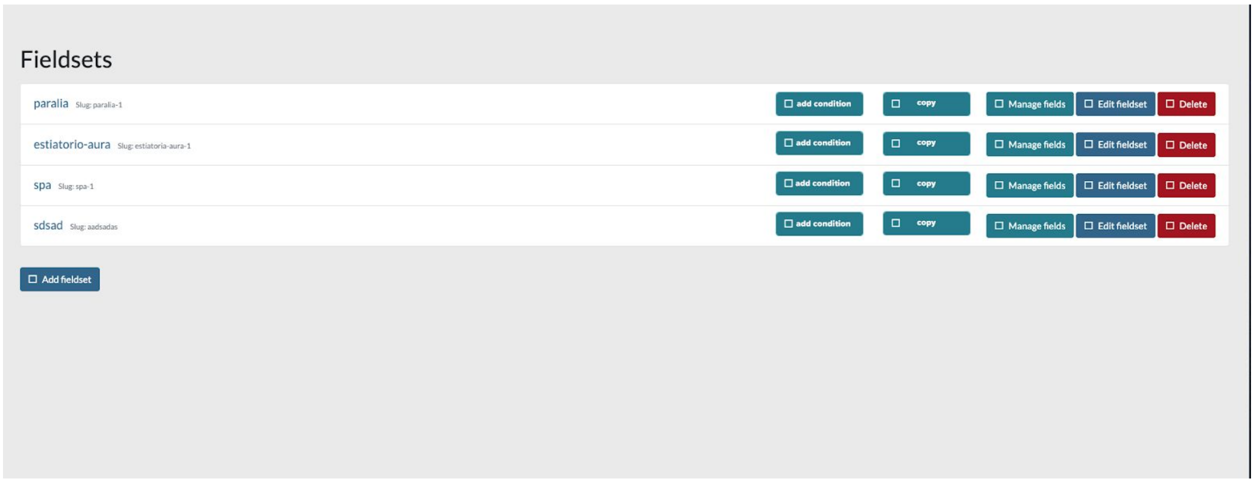
Email

Password

Figure 35. Create a user

7.5 Fieldsets page

When the administrator enters this section of the application, a list of questionnaires will appear.


Fieldsets

paralia Slug: paralia-1

estiatorio-aura Slug: estiatoria-aura-1

spa Slug: spa-1

sdsad Slug: aadsadas

Figure 36. Fieldsets page

They will be given the opportunity to manage the questions of the questionnaire. More specifically, they will be able to add a condition and add, remove, copy or edit the questionnaires present. By selecting the Manage Fields options, they can further manage the questionnaires in detail, for example, to edit a specific question in the questionnaire. The Copy option provides additional functionality in managing the questionnaires.The button "add condition" provides the functionality to create a dynamic conditional question. A user is able to copy an already existing questionnaire, effectively duplicating it, and then editing some of its questions. The Edit Fieldset option allows the users to edit the name and URL path of an existing questionnaire. Finally, the delete option deletes a questionnaire from the system.

7.5.1 Add Fieldset Subpage

The Add Fieldset option enables a user to create a new empty questionnaire. Initially, the creation of the questionnaire only requires a name and a URL path. Afterwards, they can proceed to shape the questionnaire into its final form.

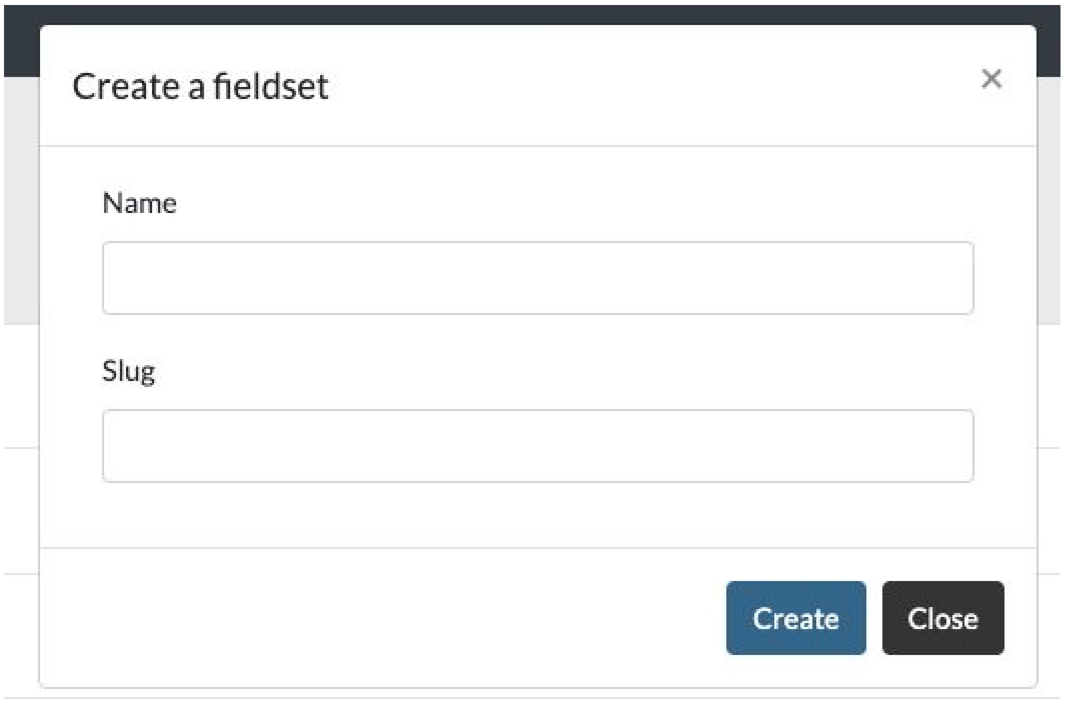
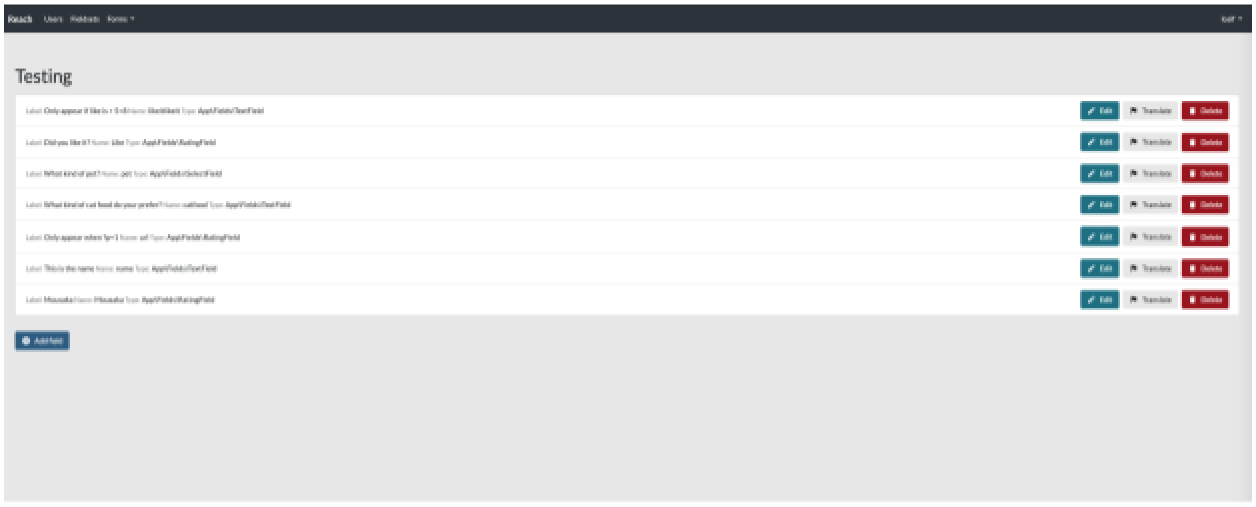


Figure 37. Create a fieldset

7.5.2 Manage Fields Subpage

By selecting the Manage Fields on a freshly created questionnaire, the user will be able to select the Add Field option to begin creating questions. A form appears with a list of fields required to be completed.

The input provided in these fields will ultimately shape the question to be added to the questionnaire. For example, a title of the question is needed, as well as a name. Furthermore, the creator needs to specify if the question will be mandatory to be answered or not. Next in line, fields that dictate the type of the question, for instance they can choose between a Text Field question, a Date Field or a Selectbox among other options. The Checkbox for the conditional field shown in the figure, enables the user to add questions that have a dependence on the customers answers to appear. For instance, a rating of Greater than 8 on a dish served will trigger a conditional question to be asked. Finally, the creator can leave additional information about a question in the form of an optional note.

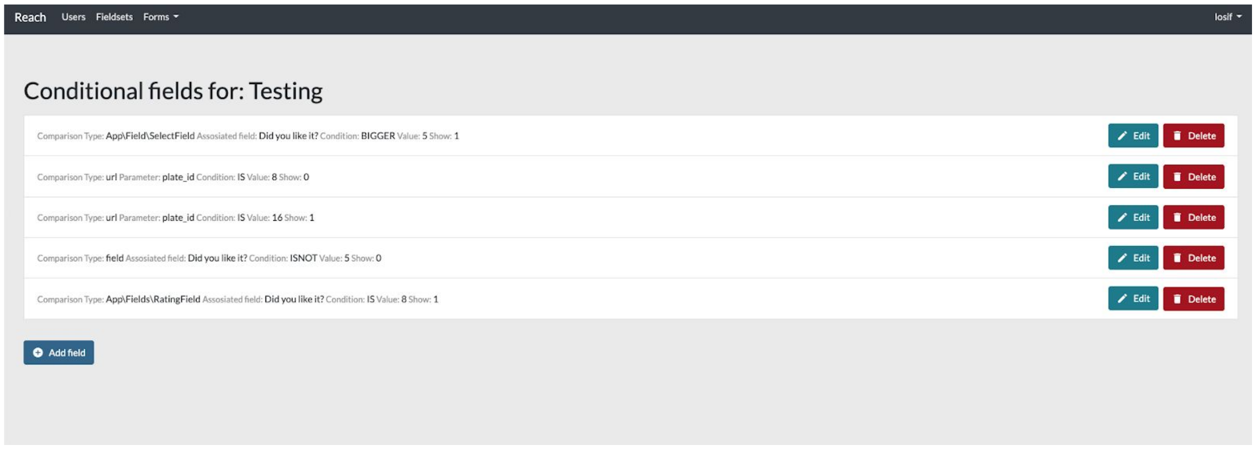


Testing

Label Only appear if like is > 5 Name [illegible] Type App\Fields\TextField
Label Did you like it? Name like Type App\Fields\RatingField
Label What kind of pet? Name pet Type App\Fields\SelectField
Label What kind of cat food do you prefer? Name catfood Type App\Fields\TextField
Label Only appear when 'p'=1 Name ad Type App\Fields\RatingField
Label This is the name Name name Type App\Fields\TextField
Label Mousaka Name Mousaka Type App\Fields\RatingField

Figure 38. Manage Fields

The button “add a condition“ gives us the opportunity to create conditions driven by the answers of our customers.



Conditional fields for: Testing

Comparison Type: App\Field\SelectField Assosiated field: Did you like it? Condition: BIGGER Value: 5 Show: 1

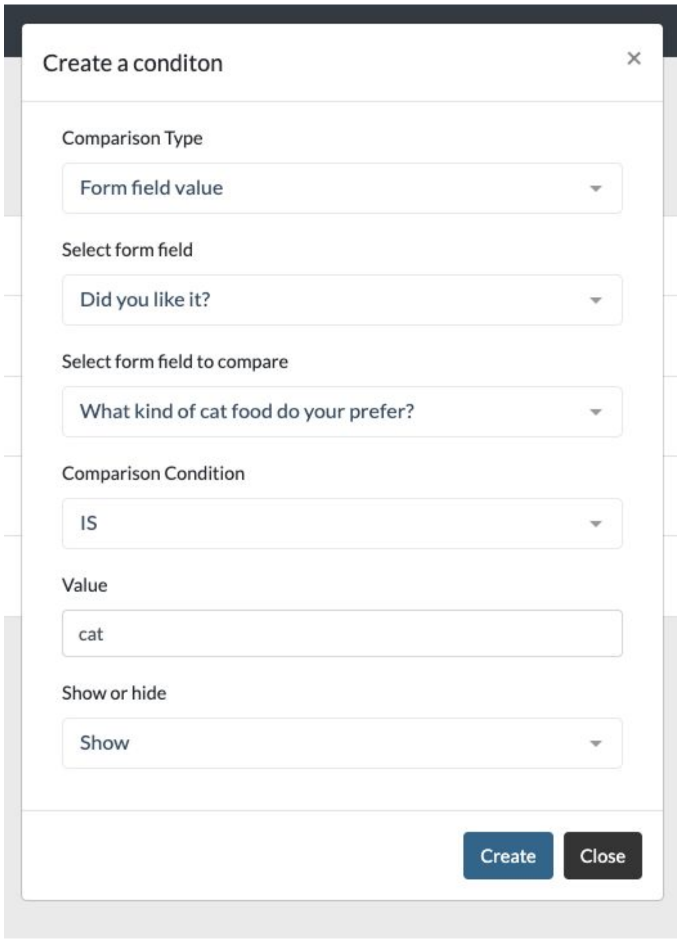
Comparison Type: url Parameter: plate_id Condition: IS Value: 8 Show: 0

Comparison Type: url Parameter: plate_id Condition: IS Value: 16 Show: 1

Comparison Type: field Assosiated field: Did you like it? Condition: ISNOT Value: 5 Show: 0

Comparison Type: App\Fields\RatingField Assosiated field: Did you like it? Condition: IS Value: 8 Show: 1

Figure 39. Conditional fields Page


Create a conditon

Comparison Type

Form field value

Select form field

Did you like it?

Select form field to compare

What kind of cat food do your prefer?

Comparison Condition

IS

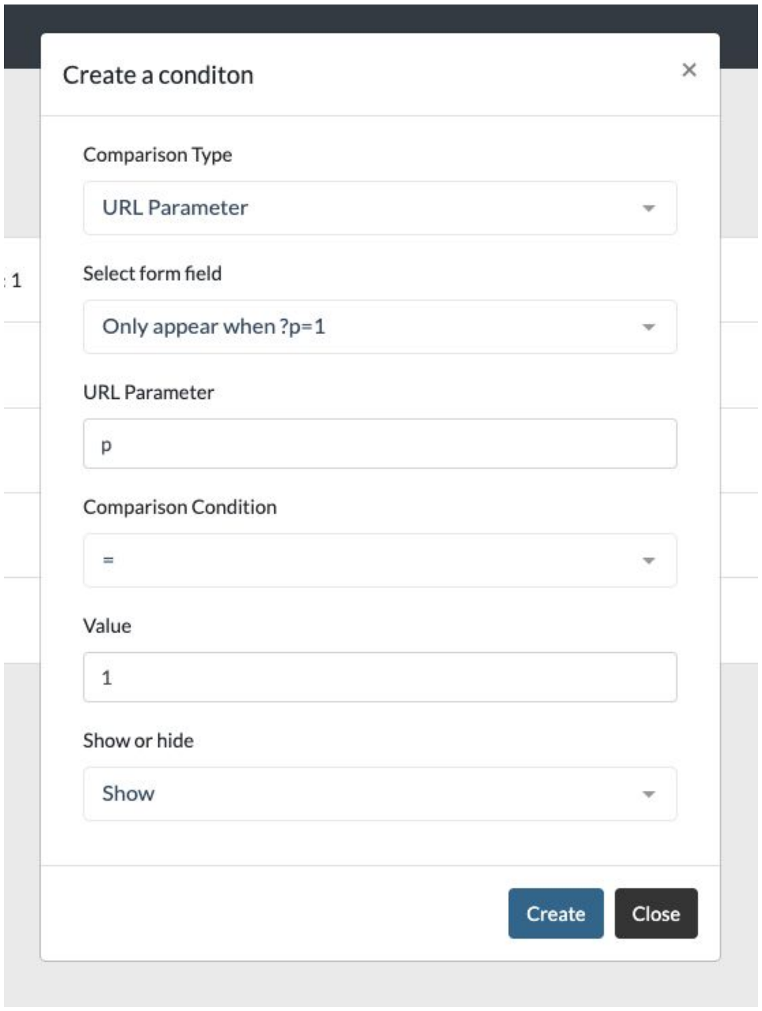
Value

cat

Show or hide

Show

Figure 40. Create a condition page 1


Create a conditon

Comparison Type: URL Parameter

Select form field: Only appear when ?p=1

URL Parameter: p

Comparison Condition: =

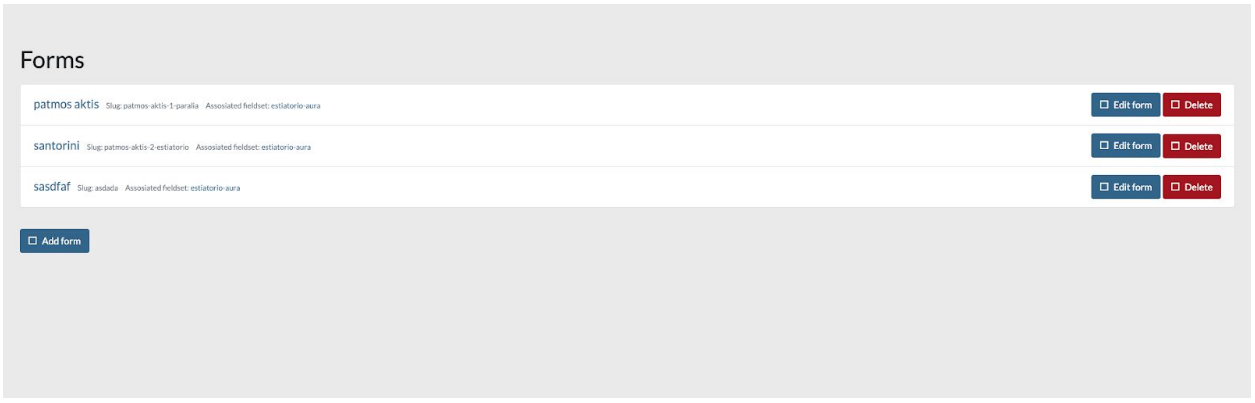
Value: 1

Show or hide: Show

Figure 41. Create a condition page 2

7.6 Forms page

In this section of the application, a list of forms will appear. Each form represents a different part of the business or even different businesses. It is possible for a business owner to have multiple departments that need to be represented in our application as separate - but connected - entities. The Forms page provides added convenience in managing the more complex business structures we might need to accommodate. An example of this is a hotel with multiple different departments, such as a pool bar, the *à la carte* restaurant, the reception and perhaps the more established fine dining restaurant. There is a need to hold different questionnaires for each department, and consequently different data collected by feedback. However, two of these departments might have a need to share a questionnaire, for instance the cocktail drinks questionnaire shared between the restaurant and the bar.


Forms

patmos aktis Slug: patmos-aktis-1-paralia Assosiated fieldset: estiatorio-aura

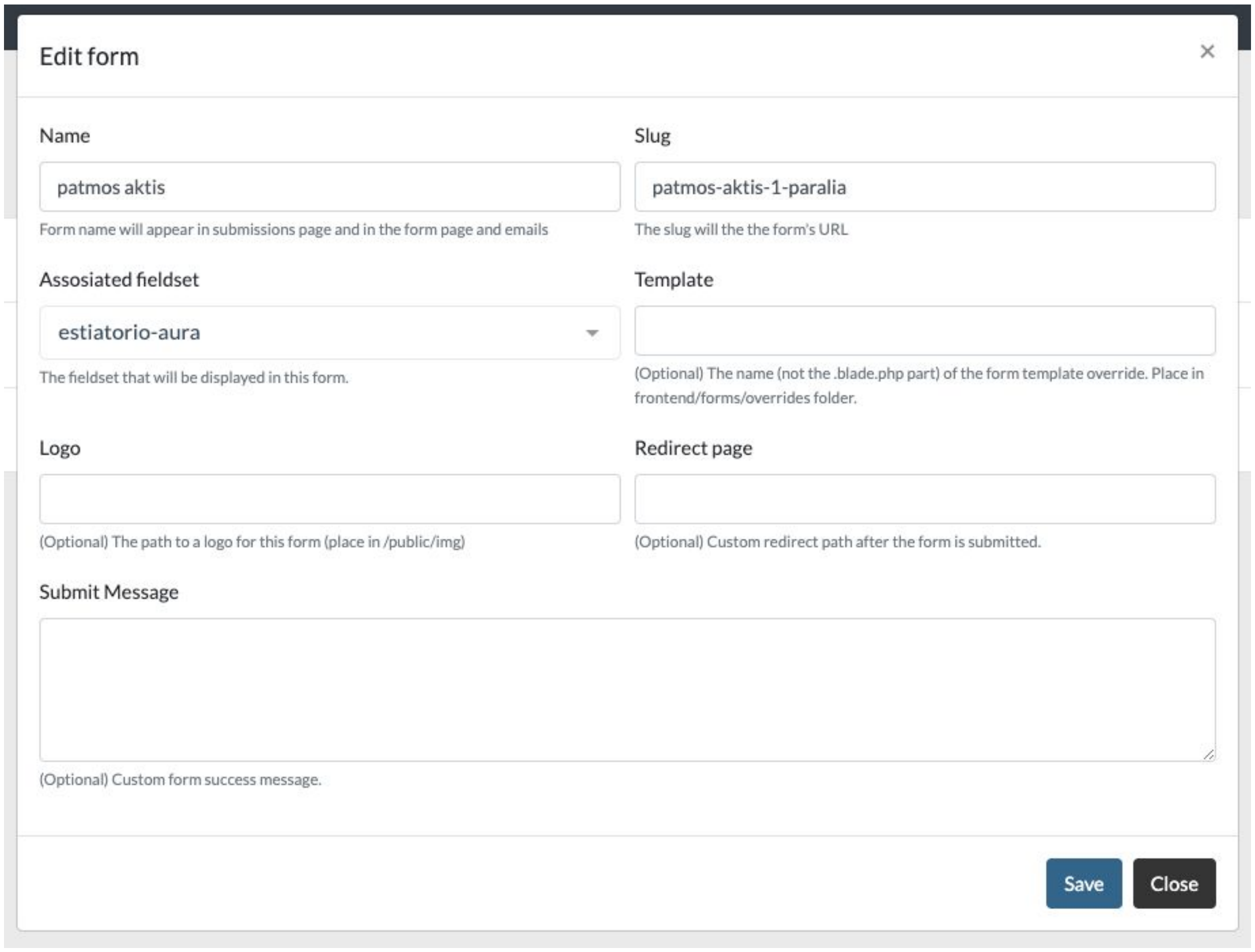
santorini Slug: patmos-aktis-2-estiatorio Assosiated fieldset: estiatorio-aura

sasdfaf Slug: asdada Assosiated fieldset: estiatorio-aura

Figure 42. Forms

Similar as before, this section provides the options to add, to edit or remove a form.

To create a new Form, one must select the Add Forms option. A collection of fields appears that require completion. First, we have the name field where the name of the Form must be given. Next, the Slug field refers to the URL path of the Form. Afterwards, the Associated fieldset drop list requires the selection of an existing fieldset. These fieldsets were described in the Fieldsets Page section. The template field is optional, and serves to provide a graphic background associated with the business, in the form of a web template. Also optional is the Logo field, that accepts an image path (.img) to be the logo of the Form. In the Redirect Page field, one can choose a redirect url path for the customer to be taken to after their questionnaire answers have been submitted. Finally, a short message to be displayed upon the questionnaire's completion can be entered in the Submit Message field.


Edit form

Name

patmos aktis

Form name will appear in submissions page and in the form page and emails

Slug

patmos-aktis-1-paralia

The slug will the the form's URL

Assosiated fieldset

estiatorio-aura

The fieldset that will be displayed in this form.

Template

(Optional) The name (not the .blade.php part) of the form template override. Place in frontend/forms/overrides folder.

Logo

(Optional) The path to a logo for this form (place in /public/img)

Redirect page

(Optional) Custom redirect path after the form is submitted.

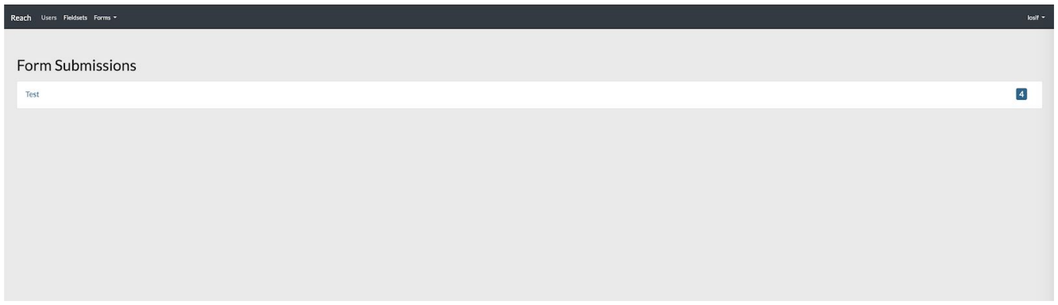
Submit Message

(Optional) Custom form success message.

Figure 43. Edit form

7.7 Form Submissions Page

The Form Submission page holds the questionnaire submissions of each form. By selecting this page a list of the Forms saved in our system is shown.

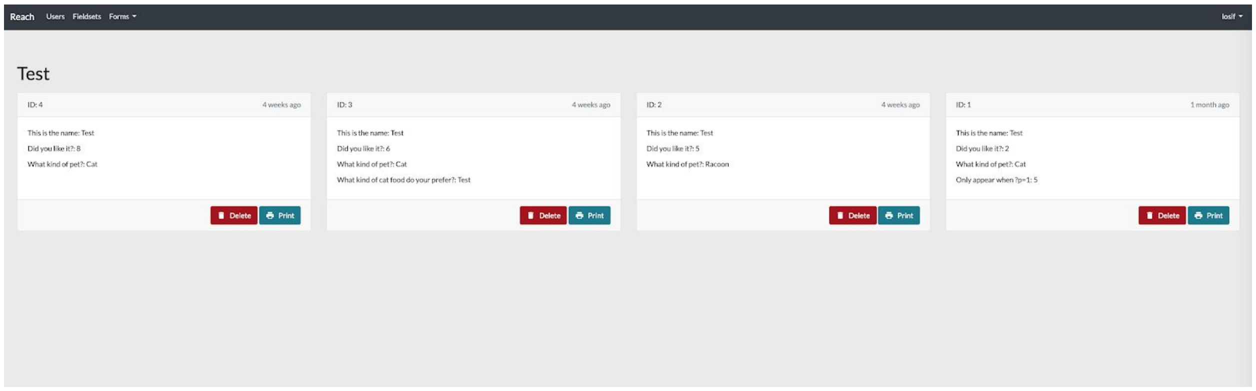


Form Submissions

Test 4

Figure 44. List of submissions

Clicking on a Form on this list will take us to the Questionnaire submissions Page. This page displays each completed questionnaire submitted in chronological order. Upon selecting one of the questionnaires on this page, one can find the questions asked and their respective answers. Additionally, the option to print or delete a unique questionnaire on this list is given.



Test

ID: 4 — 4 weeks ago
This is the name: Test
Did you like it?: 8
What kind of pet?: Cat

ID: 3 — 4 weeks ago
This is the name: Test
Did you like it?: 6
What kind of pet?: Cat
What kind of cat food do your prefer?: Test

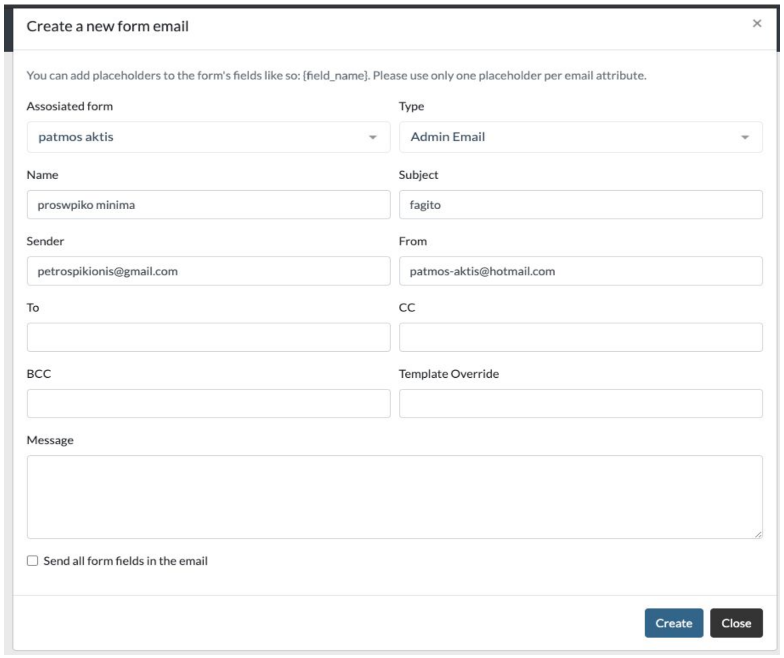
ID: 2 — 4 weeks ago
This is the name: Test
Did you like it?: 5
What kind of pet?: Racoon

ID: 1 — 1 month ago
This is the name: Test
Did you like it?: 2
What kind of pet?: Cat
Only appear when ?p=1: 5

Figure 45. Submission forms

7.8 Email Form

In this section we give the opportunity to send emails to the users of the application for a form that we have for our business or a greeting message to the customers who answered the compared form. The email form gives you the ability to choose a target group by the associated form or the type of your email


Create a new form email

You can add placeholders to the form's fields like so: {field_name}. Please use only one placeholder per email attribute.

Assosiated form: patmos aktis	Type: Admin Email
Name: proswpiko minima	Subject: fagito
Sender: petrospikionis@gmail.com	From: patmos-aktis@hotmail.com
To:	CC:
BCC:	Template Override:

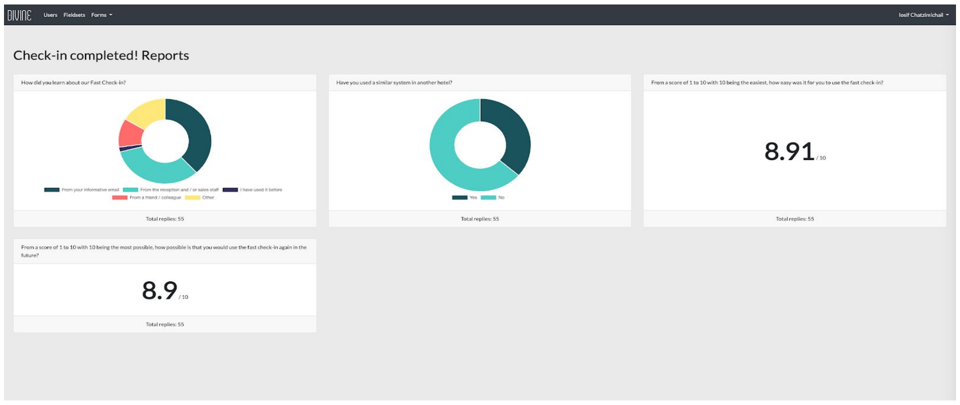
Message:

☐ Send all form fields in the email

Figure 46. Email forms

7.9 Reports

The reports page is the designated page on Data presentation of the feedback provided. As we discussed in 8.5.2 there are multiple types of questions in our questionnaires. Each type of question is linked to a different graphical technique of illustrating the data.


Check-in completed! Reports

How did you learn about our Fast Check-in?

Total replies: 55

Have you used a similar system in another hotel?

Total replies: 55

From a score of 1 to 10 with 10 being the easiest, how easy was it for you to use the fast check-in?

8.91 / 10

Total replies: 55

From a score of 1 to 10 with 10 being the most possible, how possible is that you would use the fast check-in again in the future?

8.9 / 10

Total replies: 55

DIVINE
Users
Fieldsets
Forms
Iosif Chatzimichail
From your informative email
From the reception and / or sales staff
I have used it before
From a friend / colleague
Other
Yes
No

Figure 47. Reports

7.10 Evaluation Indoors and Outdoors

The final system was evaluated by the managers of 20 food and beverage businesses located in Kos Island, during the Summer. The 20 managers evaluated the walkthrough of the application and the system itself using the "Think Aloud" methodology and took notes of their findings. The users found the application very useful and innovative but not user-friendly oriented. We conducted a survey with 20 participants, business owners in the field. At the beginning we asked some general questions about our product. ν Questionnaire targeted business owners in the Food and Beverage industry. Sample size was 20 business owners.

1) Are you confident in your ability to determine which dishes/products are your business's best sellers?
a. Very confident
b. Mildly confident, not sure
c. Not confident at all
Answers : a) 80% b) 20% c) 0%

2) Do you have time and will to gain feedback from your customers on a daily basis?
a. Absolutely
b. Yes, i make a good effort to
c. I would like to but i have no time
d. I haven't thought about it
e. I am not interested
Answers : a) 35% b) 45% c) 40%

3) Would you like to be the highest ranked business on Tripadvisor?
a. Yes, the exposure would be great
b. I am not interested
c. Not sure/ Don't know
Answers : a) 100% b) 0% c) 0%

4) How important are public relations during the tourist season to you?
a. Of utmost importance
b. Quite important
c. Not that important
Answers : a) 85% b) 15%

5) Would you show much regard to customer ratings of your business?
a. Yes
b. No
c. Not sure
Answers : a) 80% b) 5% c) 15%

At this point an in depth presentation of our application was given to all candidates, including a description of its functions and services. Then, after explaining in full detail the implementation of our idea, the procedure and how it will work, we proceeded to more targeted questions.

6) Would you be interested in an application that plots customer given feedback into statistical graphs and charts? You may select more than one answer.
a. Yes, if it was easy to use
b. Yes, if it is not too time-consuming for me and my staff
c. Yes, if it is not too expensive
d. Yes, if customers don't find it too inconvenient
e. Yes, if the results are accurate
Answers : a) 19 participants, b) 18 participants, c) 20 participants, d) 19 participants, e) 20 participants

7) Would you be willing to create the questionnaires given to your customers yourself
a. Yes
b. I don't know if i would be interested in purchasing such an app
c. Different answer
Answers : a) 20% b) 75% c) 5%

8) If the questionnaires were automatically generated through a QR CODE generator and given to the customer to answer via tablet, i would ___.
a. Definitely use the app
b. I would use the app during not peak hours
c. maybe try it/Not sure
Answers : a) 15% b) 60% c) 25%

After collecting the answers I made a record of the candidates and which business they owned.
I found that the candidates with the highest rated business on Trip Advisors were the ones that gave the most positive answers. The top four rated candidates on Trip Advisor also owned a website for their business, were present on social media platforms and were eager to try an application such as mine for their business.
12 out of 18 candidates asked if, by using our application, customer feedback of their business

could be publicly viewed - in the same way Trip Advisor allows. Upon explaining that all answers gathered from the applications questionnaires were private they requested more information on the data collection procedure the application follows as well as any evidence that supports that customers are not significantly inconvenienced by answering a questionnaire at the end of their dining. Their final question concerned the procedure needed to be followed regarding staff handling of the application.

2 candidates did not complete the questionnaire due to no interest, and 8/20 appeared sceptical/indifferent to the application. Out of these 8 candidates, 4 of them have mediocre ratings, no business website or strong presence on social media.
We concluded that they would happily use the application in their businesses should they and their employees be sufficiently trained or should the user interface be optimised. Lastly we came to the conclusion that the application should not bother employees for more than a few seconds, it should be done in an easy way (qr code scanning or mobile application putting the id of the order) .

7.11 Evaluation and Diverse Mobile Devices

Our primary target was to create a cross-platform responsive and web-native application. Therefore testing conducted on multiple devices and all behaved similarly throughout user task completion.

7.12 Summary

Chapter 7 was dedicated to the in depth description of the User Interface. A multitude of Pages was introduced, each accompanied by a description of its functional purpose. Furthermore, we describe the networking of these Pages and the means by which they communicate. Lastly, each page paragraph includes figures depicting what the end product looks like.

Chapter 8 - Summary Conclusions

The following sections summarize the above, record the system's contribution to the areas used, and refer to future extensions. In particular, Section 9.1 is a summary of the work, Section 9.2 includes points where the system improves the existing situation and Section 9.3 deals with aspects of the system that will be improved in the future.

8.1 Summary

Our application is a platform for all business owners in the field of food and beverage. The application aims to give the owner the opportunity to better monitor the management of his business. The services of our application are based on a questionnaire system that is submitted to customers when they are ready to leave the restaurant. The questionnaire is based on 2 different types of questions. The first category consists of questions corresponding to the products ordered by the customer. The second category is general questions and concerns the quality, customer service, location, etc. through a system of questions to provide useful information about the products he sells and to improve through those informations.

The way the questionnaire is created is a process that starts from the moment the waiter writes the order to his POS. The order is sent to our system, there is a list of questions that we have made and corresponds to the foods or drinks that we want to be evaluated. Our system analyzes which foods or drinks are available in each order and passes the corresponding questions to the questionnaire that will be sent to the customer. The questionnaire is completed with the general questions that have been created by the owner in consultation with us and is given to the customer to answer using a tablet or a mobile phone. The system is a form of question and question management. It enables you to create your own questions and sub-questions according to the answers, to transfer questionnaires from one store to another, to send emails to your employees or customers. The answers are stored in our system, which are converted into graphs, statistics and evaluations and sent them to the owner at regular intervals.

8.2 Contribution

Our application will be an important tool in mapping, analysis and improvement of restaurants.

The application enables the store manager to ask for information about products, service and quality offered by his business with a more detailed philosophy and to see tangible results through the responses of his customers. An important advantage is that there is no other application that interferes at the order of products and to applying questions about them. Most tools or applications reach the level where they ask for a rating or to leave a general review for their experience in the restaurant. Those reviews cannot be analyzed to data. We need data that would provide information for improving the different aspects and services of the restaurant. There is no tool or application on the market to ask questions about specific issues that the person in charge would like to ask. This larger scale process can map the best foods by region. The customer will now be able to know where to find a specific dish. The owner will have a clear picture with data and numbers of what is happening in his restaurant.

8.3 Future expansions

This section discusses the areas in which the developed system can be improved. These points mainly concern its expansion in terms of cooperation. Our application is an innovative idea that needs a lot of improvements in the future. In the future we must make a good requirement analysis for feasibility study about our product. At the moment in our application we have created questions that we have linked to an artificial product list to test our application. The customer selects products from the list and the corresponding questions appear in the questionnaire. The application will be able to work dynamically in the future when the qr code is scanned and the dynamic url will be created.

Getting our application on the market faces a challenge; A collaboration with companies that handle the PDA - based ordering of each business needs to be achieved. Through this collaboration, PDA food/drink orders and our product lists are compared, finding common listings, and, ultimately, creating a questionnaire of products to be evaluated. If we succeed and make an agreement with a PDA-ordering system provider then our business network could be expanded on a much larger scale. If such an expansion takes place, a restructure of our business model is required. No longer will it be possible to come in face to face contact with every restaurant owner. Our application will need to integrate a mechanism that automates questionnaire creation. Modifying the interface of our application to be user-friendlier, placing security measures and introducing a usage tutorial will enable business owners to design their own feedback questionnaires, removing the application administrator from the equation. One such automated mechanism could be implemented by creating a pool of questions and popular products that questionnaire creators could mix and match. Obviously, the administrator will be able to intervene and provide assistance or verify the quality of the questionnaires created.

Future development plans involve the addition of features to be used by potential customers, integrated in the mobile version of our application. An example could be a feature that allows customers, with the application downloaded, to have their personal questionnaire made on the spot on their smartphone. An order submitted through a PDA device creates a unique number, which can be submitted in our application to create a personalized questionnaire on the customer's smartphone. Another feature could be a search option where a user could request the best restaurant options for a particular product, such as a local specialty. The implementation of this feature could include a digital map, that depicts search results with business ratings, user favorites and proximity in mind. This also provides an advertising dimension to the merits of our application; our application will only be able to show businesses we have made a deal with, thus promoting them in favour of other businesses. While a system of email marketing has been implemented, it cannot be utilized to its full extent, while the mobile version of our application is still in its development stages. Email marketing can provide many assets to business owners that choose to make use of our application. The customer - application interaction is not a one time event; If they choose to sign up, they may choose to be informed about special offers or suggestions to restaurants supported by the application.

Furthermore, the optimization of submission forms handling is a pivotal step in the future of our applications development. There is great value in the ability to sort through these forms, for example by date, effectively making the usage of our app much smoother for admins and business owners alike. Equally important is the addition of an alert function, that messages a business owner on the occasion that a very low score is given. This feature allows the owner to make amends to a displeased customer and salvage an unfortunate event. The basis of our application is the statistical analysis of customer feedback. As such, the tools necessary to aggregate the data collected into tangible statistics have already been implemented. However, refining our application requires the knowledge and experience of statisticians, who excel at compiling data, as well as the interpretation and presentation of quantitative information.

References

1. JavaScript: The Good Parts
2. Crockford, Douglas, JavaScript. The good Parts: O'Reilly Media.
3. Callum Macra, Vue.js Up & Running.Building accessible and performant web apps.2018
4. Anthony Gore, Full-Stack Vue.js 2 and Laravel 5: Bring the frontend and backend together with Vue, Vuex, and Laravel,2017
5. https://laravel.com/docs/7.x
6. https://vuejs.org/v2/guide/
7. https://laracasts.com/
8. https://www.vuemastery.com/
9. https://sequelpro.com/docs

10. https://www.statista.com/page/projectaccount_overview?kw=research&crmtag=adwords&gclid=CjwKCAjw4rf6BRAvEiwAn2Q76uYJ8W389GI7PA9XyvhyHh-F9oz70bfg3DdQ3i0APt3s7d83oxAA0RoC8NcQAvD_BwE