REPUBLIC OF ALBANIA
EPOKA UNIVERSITY

FACULTY OF ARCHITECTURE AND ENGINEERING
Computer Engineering Department

CEN 302 - Software Engineering

Epoka University
Tiranë
June 2018

Meet$_{LY}$

Software documentation done by:

Deni Daja
Klesti Kuka
Bojken Sina
Ernit Hoxha

Received by: Msc. Igli Hakrama

Table of Contents

Simple things should be simple, complex things should be possible.

THIS PAGE IS INTENTIONALLY LEFT BLANK

Executive Summary

1.1 Project Overview

Up until these days, institutions whose activity was related to terrain work, especially in Albania suffered in multiple managing areas. The whole process of assigning jobs to users, keeping track of user locations, developing employee reports, tracking their performance and making sure that the employees deliver proves to be a very challenging task for many Albanian institutions and industries. As a result of these issues, many jobs were poorly executed and documented, thus resulting in missing documentation, legal trouble and a whole set of other consequences.

Meetly, the software we are providing, aims to help the institutions that are dependent on terrain work and job distribution in general. This software will provide a set of solutions for these institutions that makes every step of their work fully digitalized and easier to execute.

1.2 Purpose and Scope of this Specification

The purpose of this software is to provide a set of solutions to the institutions that depend on terrain work, job distribution and user management. In the current state, a full lifecycle of an Albanian institution when it comes to terrain work is very unorganized and unprofessional. The administrators have no way of restraining the deadlines and requirements, and the employees have no way of proving they indeed finished the job as they were required. Job details are often overlooked and this leads to major problems in the future, which may result in the need to revisit reports which is a waste of time and energy. This software will provide a very organized, hybrid and comprehensive solution for these issues, which will increase the cooperation between administrators and their employees, strengthen the trust inside the company and save some much needed time and energy for both parts that go through the process.

2. **Product/Service Description**

The Central Inspectorate is a government institution in Albania whose aim is to improve the effectiveness and accountability of the inspection activities in Albania. One of the main duties of the state is to guarantee the implementation of mandatory rules on daily operations of businesses and individuals. These rules relate to national security, fair competition, fiscal discipline, and much more. But given the importance of the Inspectorate, their execution is often not optimal.

Meetly is a software which aims to redesign the lifecycle that the Inspectorate uses to complete their jobs, and provide new technological solutions to every step of the process. This software will be used by the Central Inspectorate but can be also used by every specific Inspectorate, as our mission is to create a dynamic software, which will prove useful to every branch of the institution.

The software aims to help every step of the inspectorate job, starting from simple job distribution to employee tracking, report generation, dynamic form filling and much more.
If we divide the inspectorate in two major subsections we are able to give a more detailed description of what this software provide for both parties.

Office Workers (Chief Inspectors, Statisticians, Economists)

- Realtime Statistics, Employee Performance, Charts and Graphs
- Location-Based Jobs, Live Feeds, Employee Locations
- Job Distributions, Dynamic Forms, Reports and much more.

Terrain Inspectors

- Job Prioritization, Calculate Route Costs, Get Navigation Directions
- Check in to the job locations
- Watch job requirements and keep the constraints

2.1 Product Context

The context of this software is related to the Central Inspectorate of the government of Albania, even though our scope is to provide a solution for every branch of the Inspectorate and not only the central organization. This solution will be used by every office employee of the inspectorate and also by the terrain inspectors in order to enhance their cooperation, make their jobs easier and make sure they make a full use of their time and energy.

2.2 User Characteristics

There are six user groups that will take advantage and use the software:

- Chief Inspectors
- Terrain Inspectors
- Statisticians
- Human Resources
- Economists and Budget Handlers
- Higher Competence Employees

a) Chief Inspectors

These users will be responsible for distributing jobs to the terrain inspectors through their module called Job Distribution. They will be provided an administrator level account that will be able to CRUD jobs and employees because often they will also undertake performance evaluations for terrain workers. These users are the backbone of the system, making sure that data is generated in the system by continuously assigning jobs to terrain workers.

b) Terrain Inspectors

These users will be responsible for answering jobs that were created by the chief inspectors. Their main system will be the mobile application, and will be only restricted to that. These employees will have a user level account and are not able to CRUD or modify anything on the system, except for watching their jobs, requesting route information and filling out the forms. These users are the main feeders of data, thus making sure that the other employee groups like Statisticians and HR have enough data to perform their jobs from the system.

c) Statisticians

These users will be responsible for studying the reports and live dashboards in order to create statistics on the terrain inspectors. They will have a user level account and will not be able to CRUD or modify the system. Their job will be restricted to watching the reports and the dashboards. Through the functionality of the Reports module, these users will be able to generate very comprehensive reports in Excel and PDF.

d) Human resources

These users will be responsible for handling the employees of the institution. They will have an administrator level account and will be able to CRUD or modify the employees of the company. Their job will be restricted to creating new users, assigning privileges to the users and adjusting their salaries and rewards. They will also have access to inspector reports in order to complete performance evaluations and adjust employee salaries accordingly.

e) Economists and Budget Handlers

These users will be responsible for observing the jobs generated and the costs/rewards of each job. They will have a user level account and will not be able to CRUD or modify anything from the system. Their job will be restricted to watching task statistics, employee salaries, the fuel consumption for every job completed and creating the appropriate reports in order to provide the institution with better future budgets and more efficiency in the long run.

f) Higher Competence Employees

These users may be employees from other institutions which lie above the inspectorate like the Albanian government, whose task is to perform regular checkups on the inspectorate, to make sure their methodology is in full compliance with the law and also perform their own statistics. These users will have a user level account so they can observer everything, but are not able to modify anything on the system.

*2.3 **Assumptions***

It is assumed that the Central Inspectorate has the right to go through all the data generated from every other branch of inspectorates according to law.

It is assumed that the data generated from the system will be fully confidential and only available to the inspectorate and/or higher state institutions which govern the inspectorate.

It is assumed that every terrain inspector is equipped with a smartphone from where they will perform every operation they are required to. If not, the inspectorate is assumed responsible for equipping every inspector with the required device. These smartphones have to be capable of receiving information from GPS satellites. The smartphones are assumed to have an IOS or an Android Operating System.

It is assumed further that the inspectors are able to use smartphones and especially the Meetly mobile application effectively and efficiently.

It is assumed that terrain inspectors need to be within a certain distance from the assigned job in order to prove their correctness and in order to fill out the required form.

It is assumed that every terrain inspector should be limited to watching only jobs assigned to him and not interfere with other inspector's jobs.

It is assumed that office employees have a web browser and an active internet connection.

It is assumed that chief inspectors are aware of the location of jobs they want to distribute to terrain inspectors.

It is assumed that chief inspectors deeply understand the issues regarding specific subjects that are to be controlled, and are able to explain the problem effectively to their terrain inspectors.

It is assumed that every completed task needs to be stored in the system for documentation and research purposes.

2.4 ***Constraints***

This system will be potentially constrained by:

- The fact that inspectors have to be equipped at all times with mobile smartphones.
- The need of a fast internet connection and strong mobile data signals.
- Having every inspector understand the way the system works and training them to use the mobile application correctly and efficiently.
- Smartphones of every inspector have to be equipped with GPS technology.
- The Google Satellite should be available at all times.

2.5 ***Dependencies***

- A specific inspector can use the mobile application only after one or more jobs have been assigned to this specific inspector. Initially, inspectors will have no jobs assigned to them. At this time they will not be able to perform any operation by using their mobile application, since there is no job to operate on.

- Initially, chief inspectors will not be able to create new jobs. If the Inspectorate has no registered inspector, the job creation steps will not be complete, because this specific job cannot be assigned to anyone.

- The economists of the Inspectorate will not be able to obtain information about the monthly fuel consumption of the inspectors if no jobs are completed through the mobile application.

- The statisticians of the Inspectorate will not be able to use the Report Module for generating report files if no job distribution and job completion is not initially performed by the chief inspectors and the inspectors. In fact, this module will be fully functional when multiple jobs are assigned and/or completed.

- The performance evaluation staff cannot do their respective jobs if inspectors have not been able to complete any job by using the system. Evaluation will be more realistic and helpful if there are lots of jobs completed.

2. Requirements

2.1 Functional Requirements

The requirement numbering follows the scheme - BR_ ##

Req#	Requirement	Comments	Priority	Date	SME Reviewed / Approved
BR_01	The system should have a web application which will be used by administrators.	This will be their main platform of operation.	3	29/03/18	Klesti Kuka/ Deni Daja
BR_02	The system should provide employees with a cross-platform mobile application.	This mobile application will allow employees to accomplish their tasks.	3	29/03/18	Deni Daja/ Klesti Kuka
BR_03	The administrator has to track employee performance at all times. He should be provided with a Dashboard where he could obtain real-time information about his employees.	This module will provide administrators with statistical data. Also, a Live-Feed section will notify administrators about jobs that are being created or completed.	2	29/03/18	Deni Daja/ Klesti Kuka
BR_04	Graphically depicted data is important for every administrator. They need to be able to obtain important information easily.	Different data and information about employee performance will be depicted in various different graphs/charts in the Dashboard Module.	2	29/03/18	Klesti Kuka/ Deni Daja
BR_05	The system should provide administrators with the ability to assign jobs/tasks to their employees.	This will be accomplished by the Map module.	3	29/03/18	Deni Daja/ Klesti Kuka

Req#	Requirement	Comments	Priority	Date	SME Reviewed / Approved
BR_06	Administrators should be able to distribute jobs/tasks on terrain.	This will be accomplished by providing them with the ability to distribute jobs/tasks over a map by assigning specific address to job location.	3	29/03/18	Klesti Kuka/ Deni Daja
BR_07	Administrators should be able to specify correct location of the job on the map by only having to know the job address.	By using Geocoding services from the Google Maps API, we will convert the job addresses specified by the administrators, into precise latitude/longitude values. The latitude/longitude coordinate system allows employees to precisely know where the job is located.	3	29/03/18	Klesti Kuka/ Deni Daja
BR_08	Administrators have to keep track of jobs that are distributed to various locations.	The map module will provide real-time markers on the job locations. The markers will be shown using the latitude/longitude coordinate system.	3	29/03/18	Klesti Kuka/ Deni Daja
BR_09	Given that there will be a considerable amount of jobs distributed on the map, the administrator should be able to differentiate between them. Especially between those markers which, depending on the zoom level of the map, will overlap with each other.	The ability to differentiate between these markers will be provided by creating maker clusters, depending on the zoom levels of the map.	2	29/03/18	Klesti Kuka/ Deni Daja

Req#	Requirement	Comments	Priority	Date	SME Reviewed / Approved
BR_10	Administrators should be able to identify every single job that is distributed over the map. They should be able to check every specific job information.	The specific job information will appear on modals when administrators click the markers. They can check who that specific job was assigned to, when its deadline is, what that particular job requests the employee to accomplish etc.	2	29/03/18	Deni Daja/ Klesti Kuka
BR_11	When creating jobs, administrators need different fields where they will input job information. A form must be provided where job information will be inserted by the administrator.	This will be accomplished by having a button that will trigger a three step modal. This modal will contain the form and the fields required to create jobs. The administrator can choose the employee that the job will be assigned to, this jobs deadline, location etc.	3	29/03/18	Deni Daja/ Klesti Kuka
BR_12	Administrators should be able to also check up on jobs and their specific answers/responses. They will be able to accomplish this on the Jobs Module.	The Jobs Module will show a tabular view of every job with its respective answer/response. If the job has not been completed yet, a simple message indicating that there is no answer for this job will be shown.	3	29/03/18	Klesti Kuka/ Deni Daja
BR_13	The ability to search for any specific job on the Job Module is something that an administrator should be provided with.	This will be provided by the search bar on the Jobs Module. Jobs will also be ordered by the most recently created criteria.	2	29/03/18	Klesti Kuka/ Deni Daja

Req#	Requirement	Comments	Priority	Date	SME Reviewed / Approved
BR_14	Administrators should be able to delete outdated jobs that are no longer needed.	They can accomplish this on the Jobs Module.	2	29/03/18	Klesti Kuka/ Deni Daja
BR_15	Administrators should have the ability to keep track of every employee.	A tabular view of every employee will be provided in the Employee Module.	2	31/03/18	Klesti Kuka/ Deni Daja
BR_16	Being able to check up on every employee information is a must. Administrators should be able to easily search for a specific employee.	On the Employee Module, a search bar will help administrators look up on a specific employee easily.	2	31/03/18	Klesti Kuka/ Deni Daja
BR_17	Employees will mostly work on terrain. That is why they are prone to use a lot of the company's resources. Some of them will have to use different means of transportation. Administrators have to be able to keep track of the fuel consumption of each employee.	Information and data for each employee's monthly fuel consumption will be added in the Employee Module. This way the administrators will be able to manage company resources better.	2	31/03/18	Klesti Kuka/ Deni Daja
BR_18	The company may have a lot of employees. In this case the tabular view of employees should not overflow. Pagination will help administrators to slide through employees with ease.	Pagination for the employees will be implemented in the Employee Module.	2	31/03/18	Klesti Kuka/ Deni Daja

Req#	Requirement	Comments	Priority	Date	SME Reviewed / Approved
BR_19	Administrators have to be able to perform different operations over their employees. Adding new employees, or editing employee information and also deleting existing ones is a power that administrators have to be provided with.	The ability to perform these operations over the employees will be provided in the Employee Module.	3	31/03/18	Klesti Kuka/ Deni Daja
BR_20	Administrators should be able to obtain different reports based on their employee's performance and activity. This will be provided in the Reports Module.	In this module administrators can generate different Excel or PDF files based on employee performance or activity.	2	31/03/18	Deni Daja/ Klesti Kuka
BR_21	In the web application there is a settings/profile page provided for the administrators. Here they can update their settings and their user information.	This functionality is provided by the Setting Page.	2	31/03/18	Deni Daja/ Klesti Kuka
BR_22	The mobile application will allow each user to have its own account.	Employees will operate from this account at all times.	3	31/03/18	Deni Daja/ Klesti Kuka
BR_23	The users will be able to check their assigned jobs on the Map Module of the mobile application.	The Map Module of the mobile application will distribute markers for each job assigned to that specific employee.	3	31/03/18	Klesti Kuka/ Deni Daja

Req#	Requirement	Comments	Priority	Date	SME Reviewed / Approved
BR_24	Employees should be able to distinguish between each active job assigned to them. They should also be able to obtain job information directly from the map.	By taping each marker, the employee will be able to get job information, like job priority, deadline etc.	3	31/03/18	Klesti Kuka/ Deni Daja
BR_25	Employees can manually navigate throughout the map in order to arrive in the job location. However, they will also be provided with the ability to find the shortest path to the destination by simply taping the “Shortest Route” button on that specific job.	When this functionality is used, the shortest route from the employee’s position to the job’s destination will be highlighted. Information such as distance and the time required to get there will also be displayed.	2	31/03/18	Deni Daja/ Klesti Kuka
BR_26	The mobile application will also provide Employees with a module that will allow them to complete their assigned jobs.	In this module Employees will be able to answer to every question asked by the Administrators.	3	31/03/18	Klesti Kuka/ Deni Daja

2.2 Non – Functional Requirements

2.2.1 User Interface Requirements

The main application shall be a web application, which can be seen either with Mozilla, Chrome or Safari. The terrain inspector should install the mobile application. The main page of web application, shall a simple login interface, where it will ask the user for its Username and Password. Based on the given credentials, the necessary system constraints will apply. The user will gain access to the system, in case of proven authenticity otherwise, an error message of invalid credentials will be displayed.

As part of the system structure different system modules are included, where the user can use them, in order to ease paper work, increase work productivity and efficiency. Such modules are, dashboard module, map module, employee module, report module and the settings module.

Once the user is logged in the web application, he/she shall have access to specific modules of the system. The will redirect the user to the dashboard module. This module will provide information to the user such as, (1) the number of employees, (2) the number of active jobs, (3) the number of jobs due soon, (4) the number of jobs completed. Thus the page displayed by the system after the login interface, shall provide additional information, regarding the employees overall work performance.

The second module provided by the system is the map module. This module graphical appearance consists of a Google Maps, where the chief inspector freely manipulates the jobs. He can create a job or an activity, that needs to get completed. The chief inspector is able to create/specify jobs for the terrain inspector (2) by specifying the exact location and creating the form that needs to be filled with its specific fields.

The third module provided provided by the system is the employee module. Only the user having administrator rights can have access to such module (HR). By using this module the administrator is able to check his/her employees information. He/She will have the right to edit his/hers employees information. He/She will have the right to remove an employee from the employee list. Another module where only the administrator will have access, is the setting module. In this module he will have the right to change his/her own data credentials.

The last module provided by the system, is the reports model. This model, just like in the dashboard model will provided statistical and useful information, regarding the reports submitted by the employees. The format of such reports shall be mainly in pdf and excel format.

2.2.2 Learnability

- The application is simple to use and understand.
- The web application will come together with a PDF manual, providing a step by step information on how to effectively use the system.
- Specific error messages will be displayed, by also identifying the specific action, that caused the error.
- The application is specified for certain users, thus the system will know, when a certain action is not allowed.

2.2.3 Performance

The application, being a web application , will be stored in a web server.

The application's time of execution will depend on :

- The algorithm's efficiency for fetching data from the database.
- The users Internet connection strength.
- The servers hardware capabilities
- The operating system installed on a server.
- Third party library dependencies that need to be installed.
- The number of active user accessing the website.

2.2.3.1 Capacity

2.2.3.2

The application needs to be stored in a web server. The application itself will have a maximum size of 100 MB. The database itself will not be very large and complex. This means that the database will not occupy too much space as compared with the application. The application normally is expected to work just fine, for every user. None the less, for sake of performance and efficiency, the applications usage will be restricted, to the number of users registered on the systems database. The application should load perfectly, even by having an slow Internet connection speed. The mobile application will be deployed in the Google Play Store, it will connect to the central database and again, will have a size of no more than 100MB.

2.2.3.2 Availability

- The web application will be available for use 24/7.
- The mobile application will be available for use 24/7.
- The mobile/web application will work in an optimal manner during the working hours of the day.
- The application can be accessed and used in any geographical area, as long as the user has an active Internet connection.
- By creating separate user sessions, their overall work efficiency and productivity will not decrease by much, while using the application.
- Specific error messages will be provided, in case an action would cause systems fatal error.

2.2.3.3 Latency

The latency of the web application will mainly depend on:

- The Internet connection strength.
- The efficiency of the specific algorithm for fetching the data from the database.
- The size of the database.

Each module should show some specific performance , in order to provide fast,accurate, reliable and productive functions. Specific modules are expected to load within such time line:

- The login module should load within the first 100 ms
- The dashboard module should load within the first 300ms
- The map module should load within the first 1000 ms
- The reports module should load within the 1500 ms
- The setting module should load with the first 300 ms

2.2.4 Manageability/Maintainability

2.2.4.1 Monitoring

The applications user interface, will be easy and it will not provide cases that would crash the system. Necessary actions for any of error will be taken. The login interface needs a (1) Username and (2)Password as input. These two input data must be valid input data. The user will log in the system, in case the user has entered valid credentials, otherwise an error message of "Invalid Credentials" will be displayed.

The employee module will order the first ten employees. Within the module, a search bar is included, in order to fetch an employee having some specific credential. Employees can be searched by : (1) name , (2) email and (3) username. Placing other type of data information will result in an empty table. Also within the employee module, a failure message is included,for modifying the employees data. A failure condition is included whenever the administrator tries to change the password. If the administrator tries to place information which is not compatible with the data in the database, then an message alert will be shown. If the changed password does not match with the confirmed password then no changes are made and an alert message will be shown.

The settings module will hold the administrators personal information. Within this module the users can change their credentials. If the new data is not the same as the data type specified by the database, then an error message alert will be displayed. If the changed password does not match with the confirmed password then no changes are made and an alert message will be shown.

2.2.4.2 Maintenance

In case the system crashes, the application is going to restart. During this process, the application will redirect the user to the page where the crash occurred. In case the application did not restart correctly this two cases should be taken into consideration:

- The server storing the web application is already down and a restart is already needed.
- Contact the maintenance department, already being informed with the structure and algorithm of the application, in order to fix and make the application accessible.

The application is recommended to be stored in Linux Web Server. It is recommended to install centOS web server. In order to allow the application to use the database, it is required to install Symfony's Doctrine and its components, by especially including the doctrines migrations. It is recommended to install composer, since it responsible for updating and installing any Symfony component.

2.2.4.3 Operations

The operations mentioned below will be provided to the user of the web application:

- Add a job in Google Maps for a terrain inspector.
- Create a form that the inspector needs to complete, when finishing the specific job.
- Define the structure of the form.
- Define the nature of the questions in the form.
- Define the nature of the answer.
- Define the importance of the job.
- Define the time duration for the specific job.
- Users will have the authority to change his/her own credential data.
- CRUD functionality for employees.
- Will have the authority to add or remove new employees.
- Will have the authority of checking and evaluating employees performance.

2.2.5 Systems Interface/Integration

The database management system is the most fragile part of the application. Its access and implementation will be only provided to the IT department. The user is not allowed to make any sort of changes to the database. The application will perform this job for the user.

2.2.5.1 Network and hardware interfaces

The application being a web application needs to be stored in a web server, so that the browser user agent would be able to create a TCP connection with server. The server should function with centOS web server. The Doctrine ORM, which deals with databases needs to get installed for proper functionality.

2.2.6 Security

The application shall be developed in such way, where occurring error will be non-existent. The format of the forms are already predefined along with fields, so in this case no errors will occur. Measures are taken though, when adding new employees or when changing the employees data.

2.2.6.1 Protection

Matching function are checked when adding a new employee or while changing its credentials. The specified functions are checked:

- A function that will check for a valid email address
- A function that will check for a valid name
- A function that will check for valid name
- A function that will check for valid password

2.2.5.2 Authorization and Authentication

In this part of the software this kind of features are added:

- A function that checks for valid credentials
- A function for searching employees based on some input
- Empty message in the case that no employee is found
- Provide list of employees completing a specific report
- Provide list of employees completing a report during a specific time period
- List reports based to some criteria

2.2.6 Data Management

The database that the application will be using, will contain this kind of possible information:

- users(id,username,name,password,surname,birthday,email,phone_nr,is_admin etc)
- jobs(id,user id,title,description,deadline,priority,reward etc)
- answers(id,answer_content,answer_time etc..)
- privileges(id,privilege_name..)
- notifications(id,notification_message,notificati on_time)

2.2.8 Standard Compliance

The application will be developed in such way that will respect the rules and regulations determined by "Inspektorjati Punes", in order to provide higher proficiency and productivity in their job.

2.2.9 Portability

Portability it is not an issue. The application will be accessed as long as you have internet connection. The web application can be accessed by using either a computer or mobile phone.

2.3 Domain Requirements

- The functionality of different system users should be in "mutual exclusion" to respect the order of jobs of the Inspectorate in real life. Chief inspectors should distribute jobs, HR should handle employees and Statisticians/ Economists should create reports for the respective duties.

- The locations and data generated by the terrain inspector should be only read by authorized people in order to not have any legal implications.

- Inspectors cannot start answering or accomplishing their job from their mobile application, if they are not close to the location of that job. In order to be able to begin completing the job requirements, the inspector should at first "check-in" at the job location.

4. Software Design/Diagrams

4.1 User Scenarios

Number	User Story Name	Description
1.	Web Successful Login	Web user successfully logs in with username and password
2.	Web Failed Login	Web user fails to login with his/her credentials
3.	Chief Inspector watches Terrain Stats	Chief inspector from the dashboard watches information about terrain inspectors
4.	Chief inspector watches Real Time Feed	Chief inspector from the dashboard stays tuned on the systems real time feed and notifications
5.	Chief inspector successfully creates job	The chief inspector creates a terrain job using the 3-step procedure
6.	Chief inspector error while creating job	The chief inspector does not fulfill the criteria for creating a new job and an error occurs
7.	Chief Inspector watches job distribution	The inspector opens the “Map” module and watches the job distribution on real time
8.	Chief inspector watches job information	The inspector opens the “Map” module and watches the job information when clicking on markers
9.	Chief inspector deletes job	The inspector opens the “Map” module, clicks on a marker and presses “Delete” button to delete job
10.	Chief inspector watches a list of jobs	The inspector opens the “Job” module and a list of distributed jobs appears

11.	Chief inspector deletes a job(2)	The inspector opens the “Jobs” module, and presses “Delete” button on a specific job to delete it
12.	Chief inspector watches job answers	The inspector opens the “Jobs” module and if any job is answered, he can open the respective answer
13.	Chief inspector searches for a job	The inspector opens the “Jobs” module and searches for a specific job with a keyword
14.	Statisticians utilize the dashboard	Statisticians after logging in use the dashboard information for performing their job
15.	Statisticians generate reports	Statisticians after logging in use “Reports” module and generate reports on jobs
16.	Human Resources utilize the dashboard	Human Resources after logging in use the dashboard information for performing their job
17.	Human Resources generate reports	Human Resources after logging in use “Reports” module and generate reports on employees
18.	Human Resources deletes an employee	Human Resources after logging in use the “Employees” module to delete a user
19.	Human Resources updates an employee	Human Resources after logging in use the “Employees” module to modify user information
20.	Human Resources creates a new employee	Human Resources after logging in use the “Employees” module to create a new user
21.	Users update their profile information	Logged in user opens the “Settings” module and updates personal profile information
22.	Users sign out of the system	Logged in users click the “Logout” button and are securely redirected to login page

23.	Mobile Successful Login	Mobile user successfully logs in with username and password
24.	Mobile Failed Login	Mobile user fails to login with his/her credentials
25.	Terrain Inspector watches his jobs	Terrain Inspector clicks on the ”Map” module on the mobile application and watches assigned jobs
26.	Terrain Inspector watches job routes	Terrain Inspector clicks on the ”Map” module on the mobile application and watches routes to his/her jobs
27.	Terrain Inspector gets job information	Terrain Inspector clicks on the ”Map” module on the mobile application and watches all needed information for his/her jobs
28.	Terrain Inspector gets rejected to answer job	Terrain Inspector attempts to answer a job on “Jobs” module and gets rejected because of distance issues
29.	Terrain Inspector successfully answers a job	Terrain Inspector correctly attempts to answer a job on “Jobs” module
30.	Terrain Inspector logs out	Logged in users click the “Logout” button and are securely redirected to login page

4.2 User Scenarios

1. Web App Scenario – Successful Login

a. The user is asked to enter his username
b. The user is asked to enter his password
c. The user presses “Sign In” button
d. If his credentials are matched in the database, he is authorized to be redirected
e. The user gets redirected to the dashboard of the website (front page)

2. Web App Scenario – Failed Login

a. The user is asked to enter his username
b. The user is asked to enter his password
c. The user presses “Sign In” button
d. The credentials that were entered are not found in the database
e. The user is displayed an error message, “You credentials were not correct, please try again”
f. The user tries to enter the credentials again

3. Web App Scenario – Chief Inspector Checks Terrain Statistics

a. Chief Inspector is logged in the system
b. The information of the dashboard is loaded
c. The inspector carefully studies number of jobs completed and top performing employees
d. The inspector takes the appropriate decisions based on the data

4. Web App Scenario – Chief Inspector Watches the Realtime Feed Updates

a. Chief Inspector is logged in the system
b. The information of the dashboard is loaded
c. The inspector monitors the real-time notification feeds
d. The inspector takes the appropriate action when new jobs are answered

5. Web App Scenario – Chief Inspector Creates a New Job

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Map” module
c. The “New Job” button is clicked
d. The 3-step form modal is opened
e. The inspector fills the information of “Step 1: General Information”
f. The inspector fills the information of “Step 2: Dynamic Form”
g. The inspector fills the information of “Step 3: Location”
i. The inspector drags the marker to the correct position after geolocation
j. The inspector presses “Finish and Submit Job” button
k. The new job is added in the database and the page is refreshed
l. The job is visible in the map as a marker in the appropriate position and with a colour based on job priority

6. Web App Scenario – Chief Inspector Error While Creating a New Job

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Map” module
c. The “New Job” button is clicked
d. The 3-step form modal is opened
e. The inspector fills the information in all the 3 steps of the modal
f. The inspector has made an error on the input of some fields
g. The job is not created and a notification shows that tells the Inspector the error that was present

7. Web App Scenario – Chief Inspector Watches Job Distributions

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Map” module
c. The user zooms out in the map
d. Marker clusters are created appropriately
e. The clusters show the number of jobs in a specific zone
f. The inspectors understand the general job distributions based on location

8. Web App Scenario – Chief Inspector Watches Job Information in Map Module

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Map” module
c. The inspector clicks on a marker
d. The view is zoomed into the marker and a modal opens
e. The modal shows all the information of that specific job

9. Web App Scenario – Chief Inspector Deletes Job from Map Module

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Map” module
c. The inspector clicks on a marker and a modal opens
d. In the end of the modal, the user clicks on “Delete Job” button
e. The job is removed from the database and the page is refreshed
f. The marker is no longer in the map

10. Web App Scenario – Chief Inspector Watches Jobs on Job Module

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Jobs” module
c. A list of tabbed cards is listed in this page
d. Each card represents a job and its answer if present
e. The chief inspector can carefully watch at job information

11. Web App Scenario – Chief Inspector Watches Job Answers on Job Module

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Jobs” module
c. A list of tabbed cards is listed in this page
d. Each card represents a job and its answer if present
e. The chief inspector can open the “Answer” tab on each job
f. If the answer is complete, the answer information will show

g. If the answer is not yet completed, a message indicating that the job is not complete is shown

12. Web App Scenario – Chief Inspector Deletes a Job on Job Module

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Jobs” module
c. A list of tabbed cards is listed in this page
d. Each card has a “Delete” button in the end of the card
e. The chief inspector presses the “Delete” button on a specific job
f. A confirmation modal is shown to the inspector
g. The inspector confirms with a “Yes”. The job and its respective answer is deleted and the page is refreshed

13. Web App Scenario – Chief Inspector Searches for Job in Job Module

a. Chief Inspector is logged in the system
b. Chief inspector opens the “Jobs” module
c. A search bar is present in the beginning of the page
d. The inspector starts writing keywords on the search bar
e. The results constantly filter out the job list based on the keyword

14. Web App Scenario – Statisticians use the Dashboard

a. The statisticians are logged in the system
b. The statisticians open the “Dashboard” module
c. The statisticians take notes on dashboard information including employee performance, jobs near deadline and jobs generated to create their reports

15. Web App Scenario – Statisticians generate Reports on Employees and Jobs

a. The statistician is logged in the system
b. The statistician opens the “Reports” module
c. The statistician chooses one of the many report types by filling out required information
d. The reports are generated in PDF and the statistician can then download the report

16. Web App Scenario – Human Resources use the Dashboard

a. The HR employee is logged in the system
b. The HR employee opens the “Dashboard” module
c. The HR employee focuses on inspector performance information of the dashboard
d. The HR employee takes the appropriate actions

17. Web App Scenario – Human Resources generate Reports on Employees

a. The HR employee is logged in the system
b. The HR employee opens the “Reports” module
c. The HR employee chooses one of the many report types for employees by filling out required information
d. The reports are generated in PDF and the HR employee downloads the report
e. The HR employee takes appropriate action on the employees who underperformed/overperformed

18. Web App Scenario – Human Resources deletes an Employee in the Employee Module

a. The HR employee is logged in the system
b. The HR employee opens the “Employees” module
c. A list with all the company employees is shown in the pages
d. The HR employee deletes a specific user by pressing the delete icon
e. The employee is removed from the system and the page is refreshed

19. Web App Scenario – Human Resources updates an Employee in the Employee Module

a. The HR employee is logged in the system
b. The HR employee opens the “Employees” module
c. A list with all the company employees is shown in the pages
d. The HR employee presses on edit user icon
e. The HR employee is redirected to the editing page
f. After editing the information, the employee presses the “Update” button
g. The user is redirected to the employees page, and the employee is updated in the database

20. Web App Scenario – Human Resources creates an Employee in the Employee Module

a. The HR employee is logged in the system
b. The HR employee opens the “Employees” module
c. A list with all the company employees is shown in the pages
d. In the end of the page, the user presses “Add User” button
e. The HR employee is redirected to the adding page
f. After creating the new user information, the employee presses the “Create” button
g. The user is redirected to the employees page, and the employee is created in the database

21. Web App Scenario – System Users Edit Their Profile Information

a. The system user is logged in
b. The system user opens the “Settings” module
c. The system user is shown a page that contains their profile settings
d. The users edit their profile information
e. The users press the “Submit” button
f. The information is updated in the database and the page is refreshed

22. Web App Scenario – System Users Logout

a. The system user is logged in
b. The system user presses the “Log out” button in the navbar
c. The system user is signed out securely
d. The system user is redirected to the login page
e. The system user cannot go back to the page by using back button, only by re-entering the credentials

23. Mobile App Scenario – Successful Login

a. The terrain inspector is asked to enter his username
b. The terrain inspector is asked to enter his password
c. The user presses “Sign In” button
d. If his credentials are matched in the database, he is authorized to be redirected
e. The terrain inspector gets redirected to the main page of the mobile application

24. Mobile App Scenario – Failed Login

a. The terrain inspector is asked to enter his username
b. The terrain inspector is asked to enter his password
c. The terrain inspector presses “Sign In” button
d. The credentials that were entered are not found in the database
e. The terrain inspector is displayed an error message, “You credentials were not correct, please try again”
f. The terrain inspector tries to enter the credentials again

25. Mobile App Scenario – Terrain Inspector Watches His Jobs in Map Module

a. The terrain inspector opens the Map Module
b. A map that shows all the inspectors jobs as markers is shown
c. The markers are colored based on priority
d. The terrain inspector presses a marker
e. A menu with 3 options is shown “Watch Route”, “General Information”, “Exit”

26. Mobile App Scenario – Terrain Inspector Watches Route for Job in Map Module

a. The terrain inspector opens the Map Module
b. A map that shows all the inspectors jobs as markers is shown
d. The terrain inspector presses a marker
e. A menu with 3 options is shown “Watch Route”, “General Information”, “Exit”
f. The terrain inspector clicks on “Watch Route”
g. The route distance and estimated time is shown to the inspector
h. The route is drawn on the map with a blue line

27. Mobile App Scenario – Terrain Inspector Gets Job Information Map Module

a. The terrain inspector opens the Map Module
b. A map that shows all the inspectors jobs as markers is shown
d. The terrain inspector presses a marker
e. A menu with 3 options is shown “Watch Route”, “General Information”, “Exit”
f. The terrain inspector clicks on “General Information”
g. A modal with the job information is shown to the user

28. Mobile App Scenario – Terrain Inspector Gets Rejected to Answer a Job

a. The terrain inspector opens the Jobs Module
b. A list of all assigned jobs is shown
d. The terrain inspector presses the “Answer this Job” button
e. The distance from the inspector location and job is greater than 5km, so the system does not allow it to avoid any fraud

f. The terrain inspector gets notified that he is not within the required range

29. Mobile App Scenario – Terrain Inspector Answers a Job

a. The terrain inspector opens the Jobs Module
b. A list of all assigned jobs is shown
d. The terrain inspector presses the “Answer this Job” button
e. The inspector is redirected to the answering page, where the form generated by the chief inspector is shown
f. The inspector fills out the form and presses “Submit”
g. The answer is added in the database, and the job is removed from the active jobs list
h. The chief inspectors get notified and watch the answer in the web app

30. Mobile App Scenario – Terrain Inspector Logout

a. The terrain inspector is logged in
b. The terrain inspector presses the “Log out” button in the application navigation bar
c. The terrain inspector is signed out securely
d. The terrain inspector is redirected to the login page
e. The terrain inspector cannot go back to the page by using back button, only by re-entering the credentials

4.3 Use Cases

Name	User Login
Summary	The user enters the system by entering valid credentials
Actor	Chief Inspector, Statistician, Terrain Inspector, Human Resources, Economist and Budget Handlers, Higher Competence Employees
Description	The user in order to gain access to the modules of system, needs to log in first. In order to log in it must first provide the correct username and password.
Precondition	The user who loges in, must have first an existing account, with a specific username and password.
Alternative	There are no alternative options. This is done in order to protect confidential information.
Post Condition	To gain access to the modules provided by the software system.

Name	Check Terrain Statistic
Summary	The Chief inspector checks the number of active jobs and the number of jobs completed.
Actor	Chief Inspector in this case is the primary actor.
Description	The chief it is redirected at the dashboard, once it has logged in by providing the necessary and correct credentials. From there it checks how many employees it has, how many jobs are currently and still need to be completed.
Precondition	The chief inspector like all other users needs to have an existing account, but at the same time, it must have administrator privileges.
Alternative	There are no alternative options.
Post Condition	To check status of the jobs that had been created and take necessary actions in case some jobs were not completed.

Name	Real Time Feed Updates
Summary	The Chief Inspector monitors real time feeds, generated, due to the completion of work.
Actor	The primary actor is the chief inspector
Description	The chief inspector does not only need to check for completed jobs, but must also provide real time feedback to the employee in the terrain. This is done by monitoring real time feeds generated by the employee due to the completion of a certain job.
Precondition	The chief inspector must be logged in to the system. In the system must exist active jobs.
Alternative	No time feeds will be displayed
Post Condition	The chief inspector answer in real time to the employees working in the terrain.

Name	Job Creation
Summary	The chief inspector creates a new job for a specific employee through the map module.
Actor	The primary actor is the chief inspector
Description	The chief inspector is able to create a new active job. The job has a specific priority, it is located by checking google, and it contains a form, that the employee needs to complete.
Precondition	The data information for the employee, whom an active job is created should already exist.
Alternative	Create a new active job, but with no employee mentioned.
Post Condition	The creates a new job in real time, for the employee to see.

Name	Job Deletion
Summary	The chief inspector deletes a job that already exists in the google map.
Actor	The primary actor is the chief inspector
Description	In case a certain job is canceled, the chief inspector can remove the active job from the google map
Precondition	The chief inspector must be logged in into the system. There must already be an active job in the google map.
Alternative	Mark the job as completed.
Post Condition	To delete unnecessary jobs. This will result in increase of work productivity and efficiency.

Name	Job Distribution
Summary	The chief inspector checks how jobs are distributed in a specific area location in the google map.
Actor	The primary actor is the chief inspector
Description	The chief inspector can check the job distribution in certain area location. By zooming in, the number of jobs can made visible.
Precondition	The chief must be logged in into the system. A well defined number of jobs must exist at a specific area.
Alternative	There are no alternatives.
Post Condition	Be aware about the distribution in an area. Give a work to an employee that is close by.

Name	Job List
Summary	The chief inspector can check the job list from the job module.
Actor	The primary actor is the chief inspector
Description	A list of of jobs are presented in the job module. The chief inspector can check the status of the specific job or any other additional information related with the specific job.
Precondition	The chief must be logged in into the system. A number of active jobs must already exist.
Alternative	Check the active jobs in th map module.
Post Condition	Be aware for each active job, their job status and other additional information

Name	Job Answer
Summary	The chief inspector checks if the job has been answered or not at all.
Actor	The primary actor is the chief inspector
Description	The chief inspector can check if a job has been answered through the job module. If the the job has been answered then the specific will be shown, otherwise a not completed message will be displayed instead.
Precondition	The chief must be logged in into the system. A well defined number of jobs must exist at a specific area. A job is completed or almost completed.
Alternative	Do not send an answer if a job it is not completed
Post Condition	Be aware about the completion of a job and take measures if its not.

Name	Delete Job Answer
Summary	The chief inspector deletes a job from the job module.
Actor	The primary actor is the chief inspector
Description	The list of jobs provided in the job module have a delete button. By clicking that button, a specific job in the job list can be removed.
Precondition	The chief must be logged in into the system. A number of jobs must exist. The job is answered.
Alternative	There are no alternatives
Post Condition	Keep track of active job, that do not have an answer. To remove old jobs.

Name	Job Distribution
Summary	The chief inspector checks how jobs are distributed in a specific area location in google maps.
Actor	The primary actor is the chief inspector
Description	The chief inspector can check the job distribution in certain area location. By zooming in, the number of jobs can made visible.
Precondition	The chief must be logged in into the system. A well defined number of jobs must exist at a specific area.
Alternative	There are no alternatives.
Post Condition	Be aware about the distribution in an area. Giving a work to an employee that is close by.

Name	Job Search
Summary	The chief inspector finds a specific according to a specific information data.
Actor	The primary actor is the chief inspector
Description	The chief inspector can search for a specific job, by placing some certain data information.
Precondition	The chief must be logged in into the system. The job must already exist.
Alternative	There are no alternatives
Post Condition	Fetch a specific job , when list o jobs is large.

Name	Statisticians use of dashboard
Summary	The statistician analyze the information provided by the dashboard module and create their own report
Actor	The primary actors are the statisticians
Description	The statisticians check the dashboard information. They also check the information provided by employee in order to generate their own report.
Precondition	The statisticians must be logged in.
Alternative	Inconsistencies might occur if a job gets deleted, during the moment they are generating the report.
Post Condition	Generate a result regarding work productivity and efficiency, for that specific day.

Name	Generate Report
Summary	Statisticians generate a report from the employee through the report module.
Actor	The primary actors are the statisticians.
Description	The statisticians can generate a report for the specific employee. This is done in the report module. Information like the number of jobs completed, the number of jobs active are included in the report.
Precondition	The statistician must log in. The employee must have completed a number of jobs.
Alternative	No reports can be done for a new employee.
Post Condition	Measure the work productivity and efficiency shown by the employee.

Name	Human Resources Scenario
Summary	Human resources measure the work productivity of the chief inspectors.
Actor	The primary actors are the human resources.
Description	The human resources can evaluate the work productivity and efficiency shown by the chief inspectors. They can do that by generating their own report. In their report they may include the information provided in the dashboard.
Precondition	The human resources must be logged in.
Alternative	There are no alternatives.
Post Condition	Measure the work productivity and efficiency shown by the chief inspector.

Name	Human Resources Report Module
Summary	Human resources measure the work productivity of the employees.
Actor	The primary actors are the human resources.
Description	The human resources can evaluate the work productivity and efficiency shown by the employees. They can do that by generating their own report. In their report they may include the information provided in the report module.
Precondition	The human resources must be logged in.
Alternative	There are no alternatives.
Post Condition	Measure the work productivity and efficiency shown by the employees.

Name	Human Resources Delete An Employee
Summary	Human resources delete an employee
Actor	The primary actors are the human resources.
Description	The human can remove an employee from the employee list, by clicking the remove icon in the employee module.
Precondition	The human resources must be logged in.
Alternative	There are no alternatives.
Post Condition	Removing an employee who does not work anymore.

Name	Human Resources Update An Employee
Summary	Human resources update the information regarding an employee through the employee module.
Actor	The primary actors are the human resources.
Description	The human resources can update the employees information through the employees module.
Precondition	The human resources must be logged in.
Alternative	There are no alternatives.
Post Condition	Update employee information

Name	Human Resources Create An Employee
Summary	Human resources create an employee through the employee module.
Actor	The primary actors are the human resources.
Description	The human resources can create a new employee, by clicking the add button in the employee module.
Precondition	The human resources must be logged in.
Alternative	There are no alternatives.
Post Condition	Create a new employee who was recently hired.

Name	System User Edit Information.
Summary	The user can change their profile information.
Actor	The actors are the chief inspector,statisticians, human resources,economists an budget handlers and terrain employees.
Description	All the actors that use the system can change their profile information through the settings module.
Precondition	The actors must log in.
Alternative	There are no alternatives.
Post Condition	Change specific data in their profile information.

Name	System User Log Out
Summary	The user closes her account through the log out button.
Actor	The actors are the chief inspector,statisticians, human resources,economists an budget handlers and terrain employees.
Description	All the actors that use the system can log out from their account.
Precondition	The actor or user must be logged in first.
Alternative	There are no alternatives.
Post Condition	Log out from their account.

Name	Mobile System Log In
Summary	The terrain inspector logs in into their mobile account, by placing the account credentials.
Actor	The primary actor is the terrain inspector.
Description	The terrain inspector signs in into his mobile account, by providing his username and password.
Precondition	The terrain inspector must already have a mobile account.
Alternative	There are no alternatives.
Post Condition	Terrain inspectors log in into their mobile accounts.

Name	Mobile Map Module
Summary	The terrain inspector can check his to due jobs in the map module.
Actor	The primary actor is the terrain inspector.
Description	The terrain inspector can check the jobs assign to him/her by checking the map module. The jobs assigned to the specific employee, are represented in different color, according to the jobs priority.
Precondition	The terrain inspector must be logged in. The terrain inspector must already have some active jobs.
Alternative	There are no alternatives.
Post Condition	The terrain inspector can check the job assigned to him/her, the location and the last deadline for the job to get completed.

Name	Mobile Job Route.
Summary	The terrain inspector can check the job route in the map module.
Actor	The primary actor is the terrain inspector.
Description	The terrain inspector can check the jobs assign to him/her by checking the map module. Each job has an icon, which is placed in the google map. The user clicks in the icon and the option for showing the route , is displayed.
Precondition	The terrain inspector must be logged in. The terrain inspector must already have some active jobs.
Alternative	There are no alternatives.
Post Condition	Calculate the route.

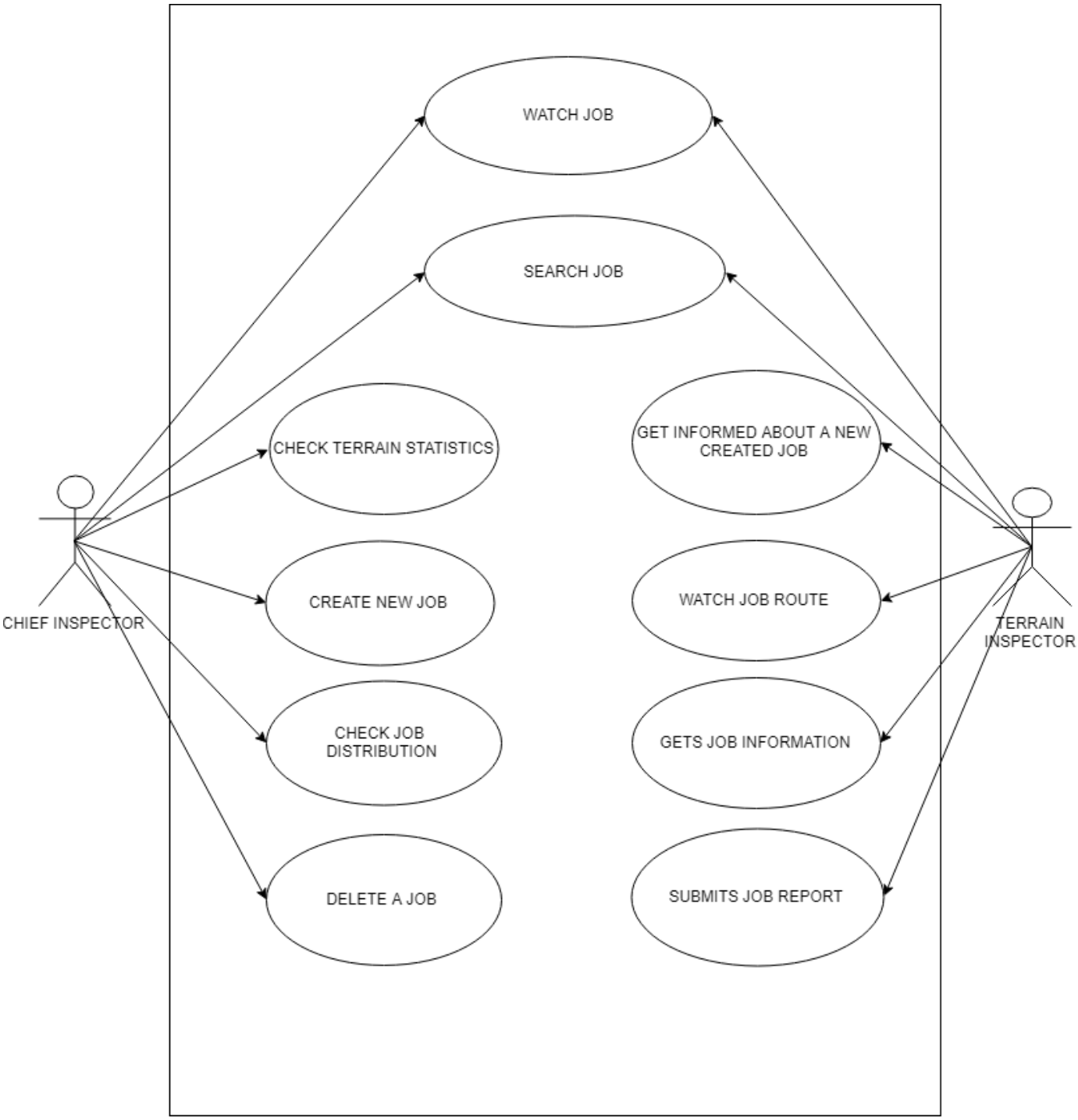
Name	Mobile Job Information.
Summary	The terrain inspector can check the job information in the map module.
Actor	The primary actor is the terrain inspector.
Description	The terrain inspector can check the jobs assign to him/her by checking the map module. Each job has an icon, which is placed in the google map. The user clicks in the icon and the option for showing the job information , is displayed.
Precondition	The terrain inspector must be logged in. The terrain inspector must already have some active jobs.
Alternative	There are no alternatives.
Post Condition	Obtain general information regarding an active job.

Name	App Job Answer.
Summary	The terrain inspector can answer to the specific job through the job module.
Actor	The primary actor is the terrain inspector.
Description	The terrain inspector sends a feedback answer, once a job gets completed.
Precondition	The terrain inspector must already have some active jobs. The terrain inspector must have completed the answer form first.
Alternative	There are no alternatives.
Post Condition	Calculate the route.

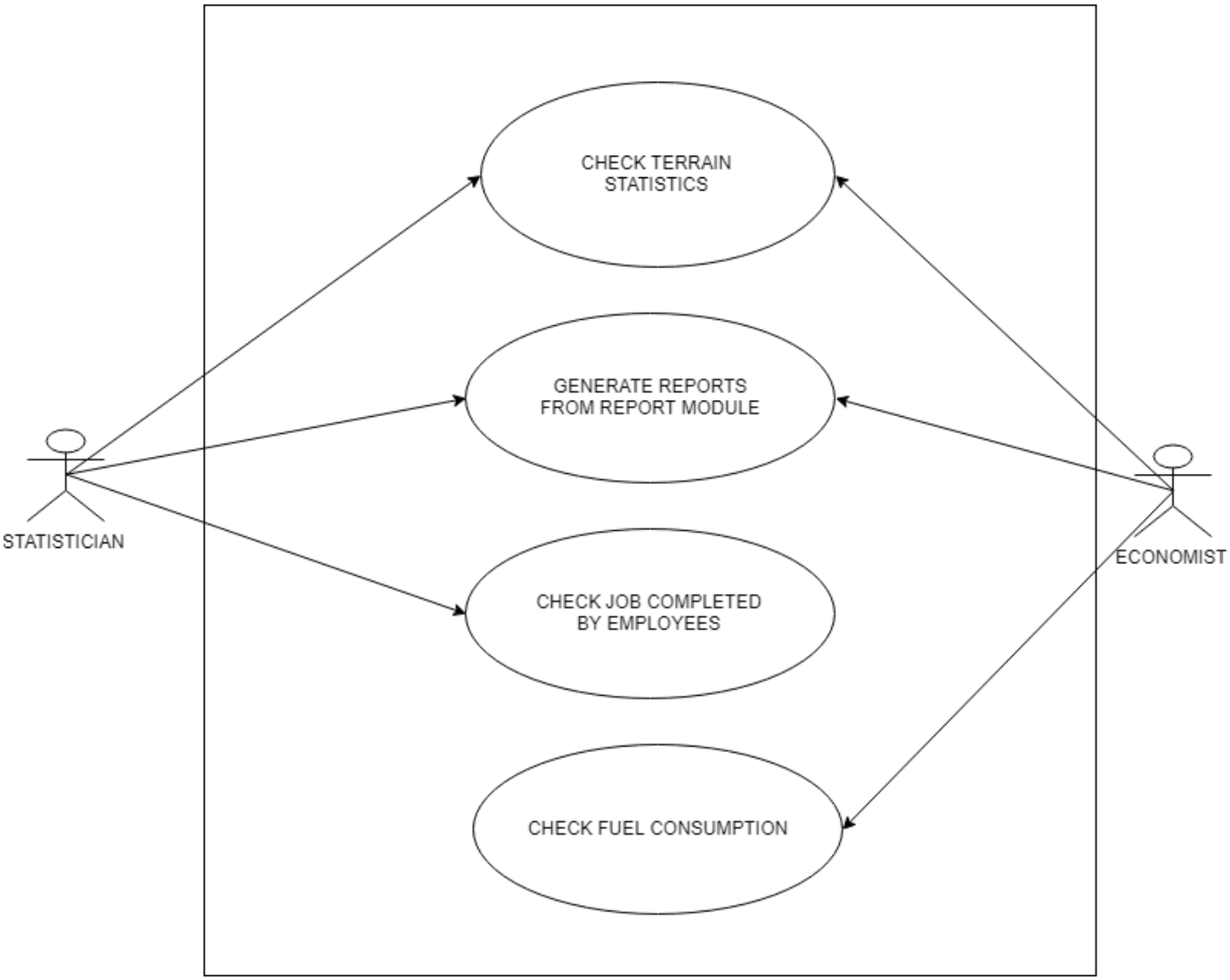
Name	App Log Out
Summary	The terrain inspector can log out from his/her mobile account.
Actor	The primary actor is the terrain inspector.
Description	The terrain inspector logs out from his mobile account, by clicking the log out button.
Precondition	The terrain inspector must be logged in first.
Alternative	There are no alternatives.
Post Condition	Log out from the account.

4.4 Use Cases Diagrams

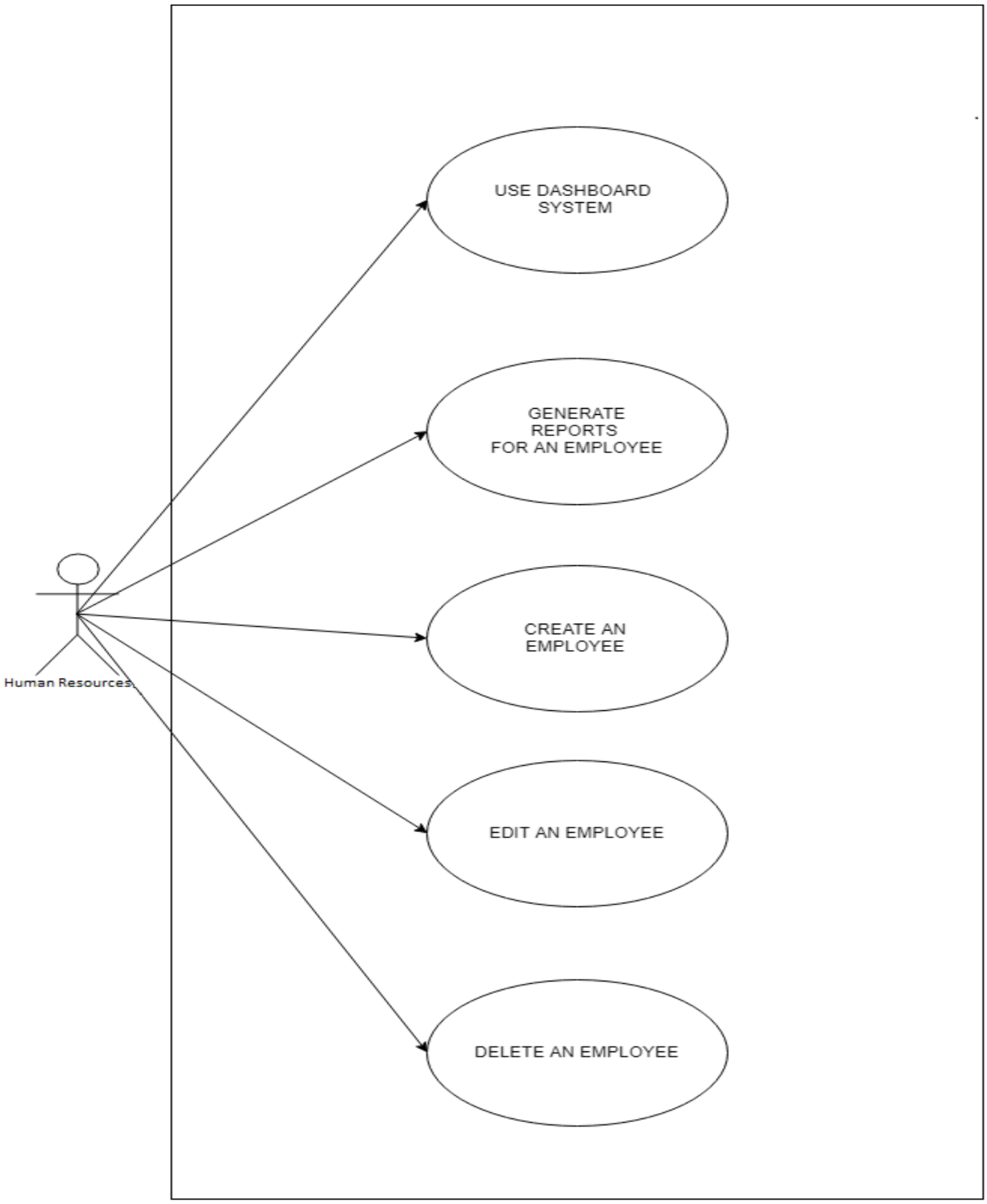
Chief - Terrain Inspector Use Case Diagram



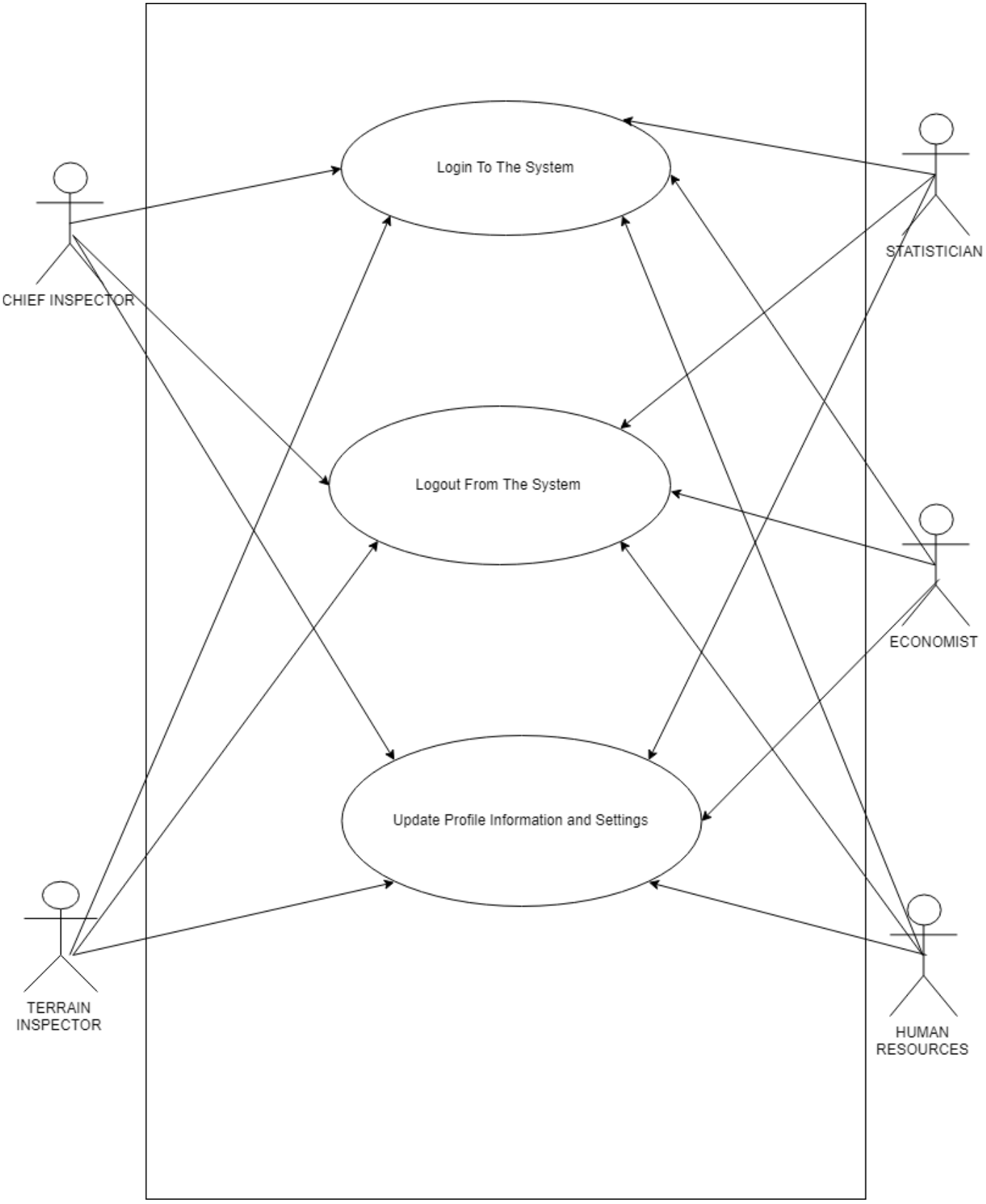
Economist - Statistician Use Case Diagram



Human Resources Use Case Diagram

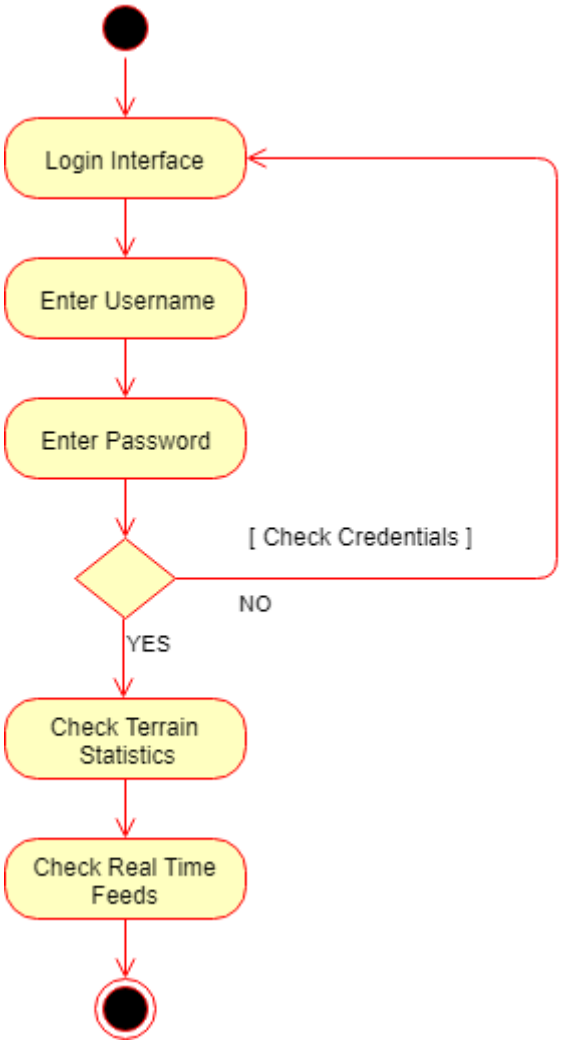


All Users Use Case Diagram

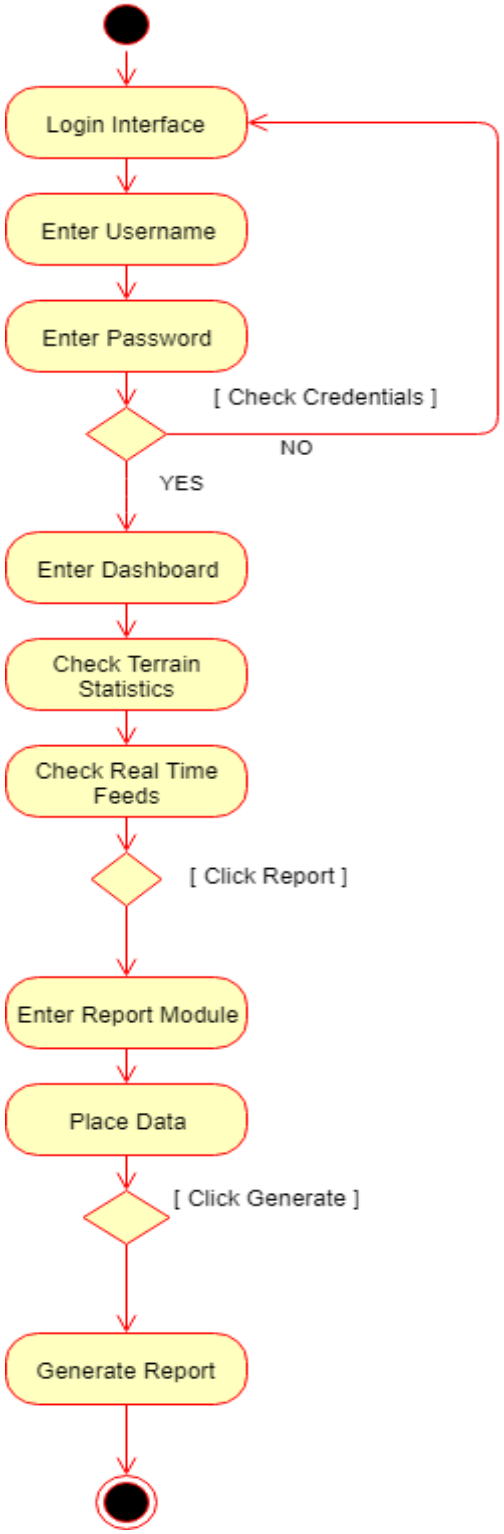


4.5 Activity Diagrams

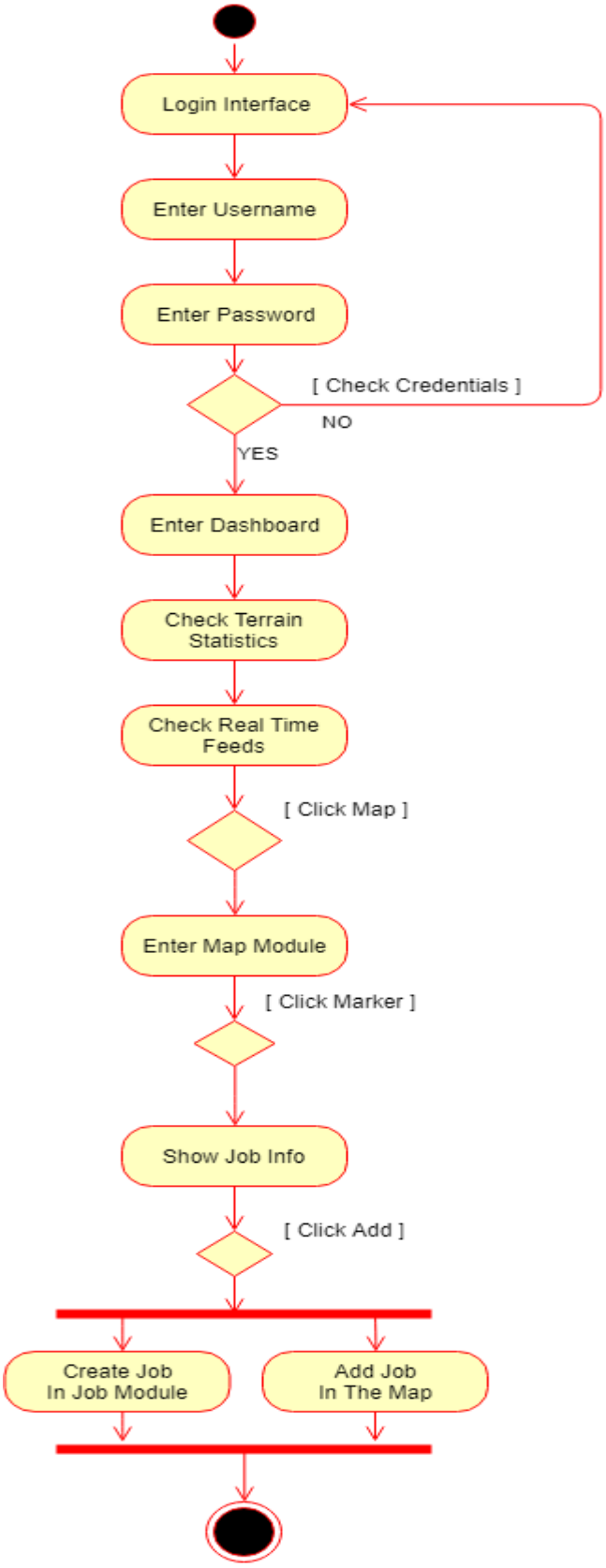
Chief Inspector Dashboard Activity Diagram



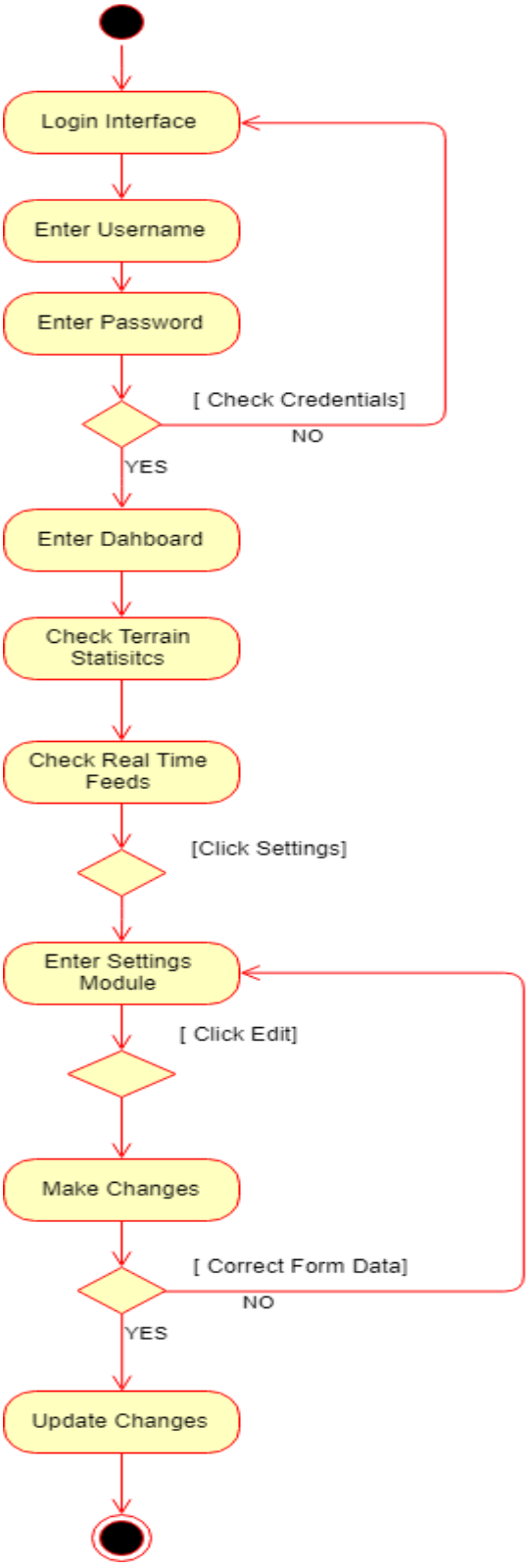
Chief Inspector Report Activity Diagram



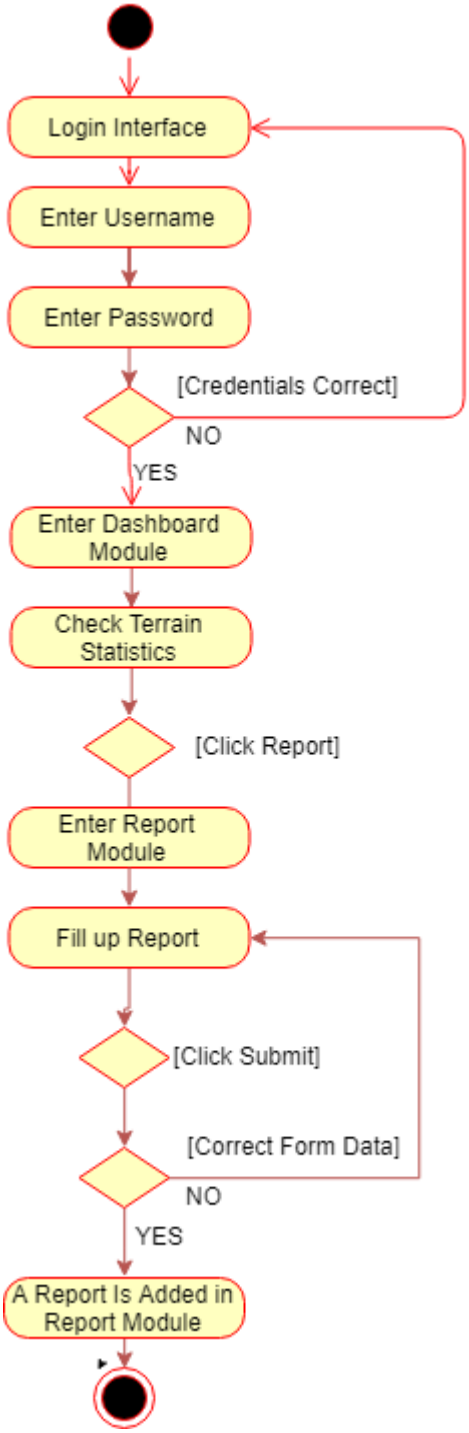
Chief Inspector Map Module



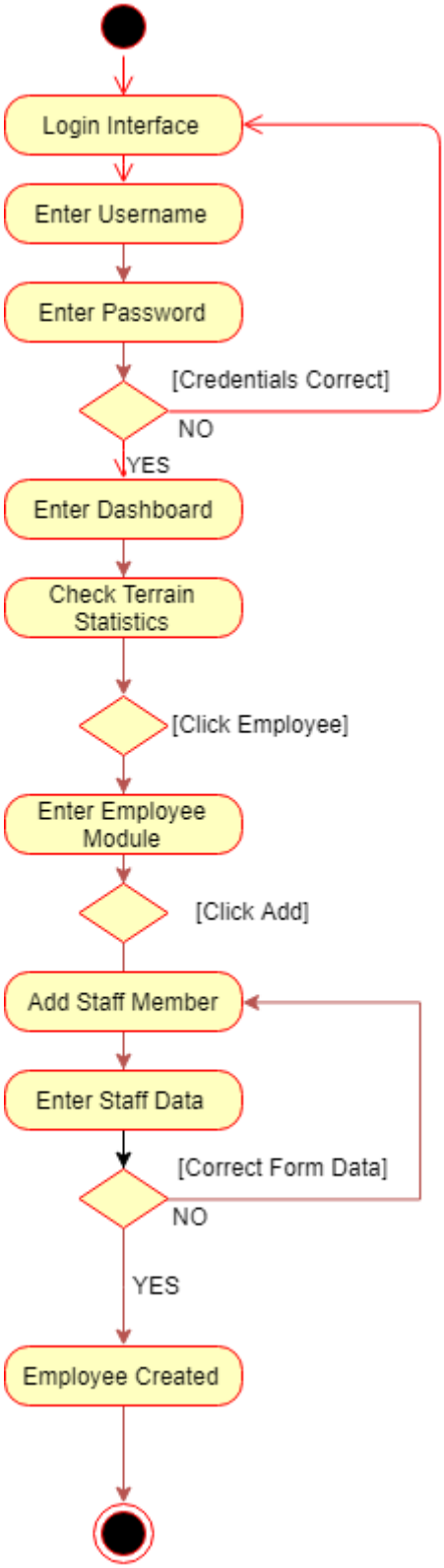
Chief Inspector Settings Module



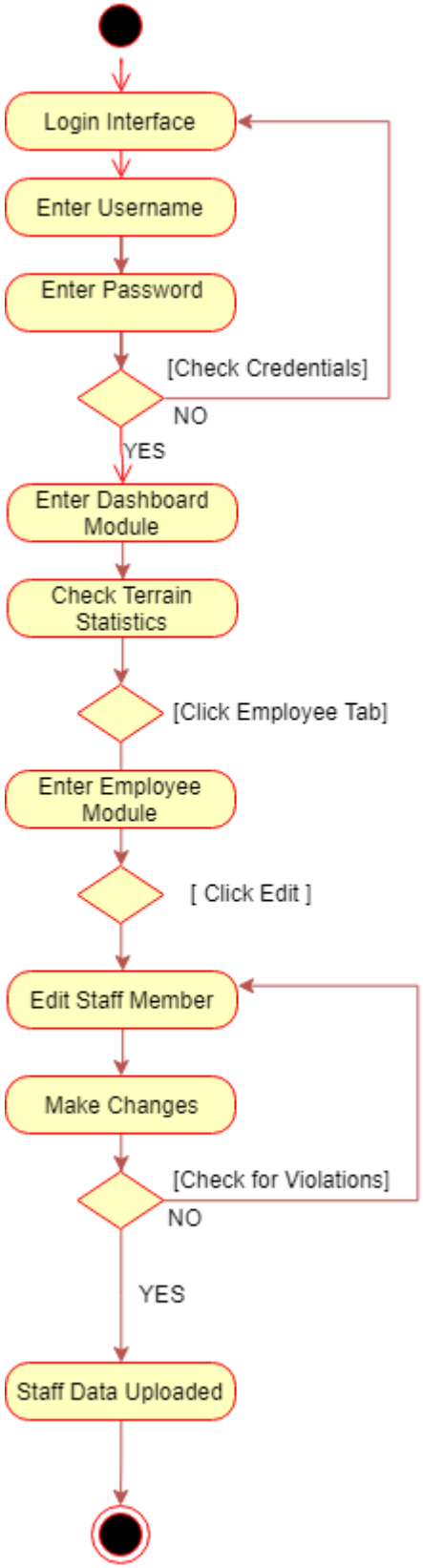
Human Resources Activity Diagram 1



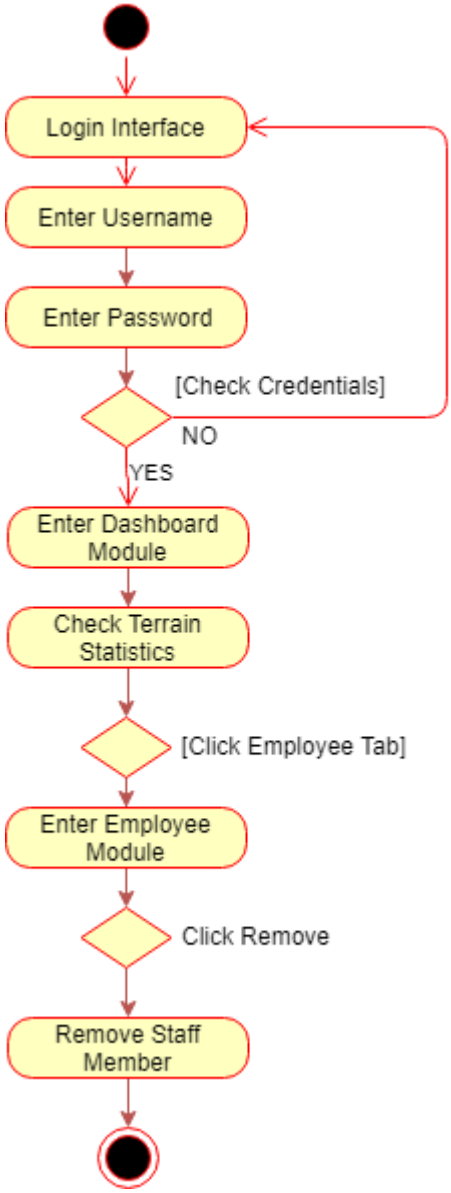
Human Resources Activity Diagram 2



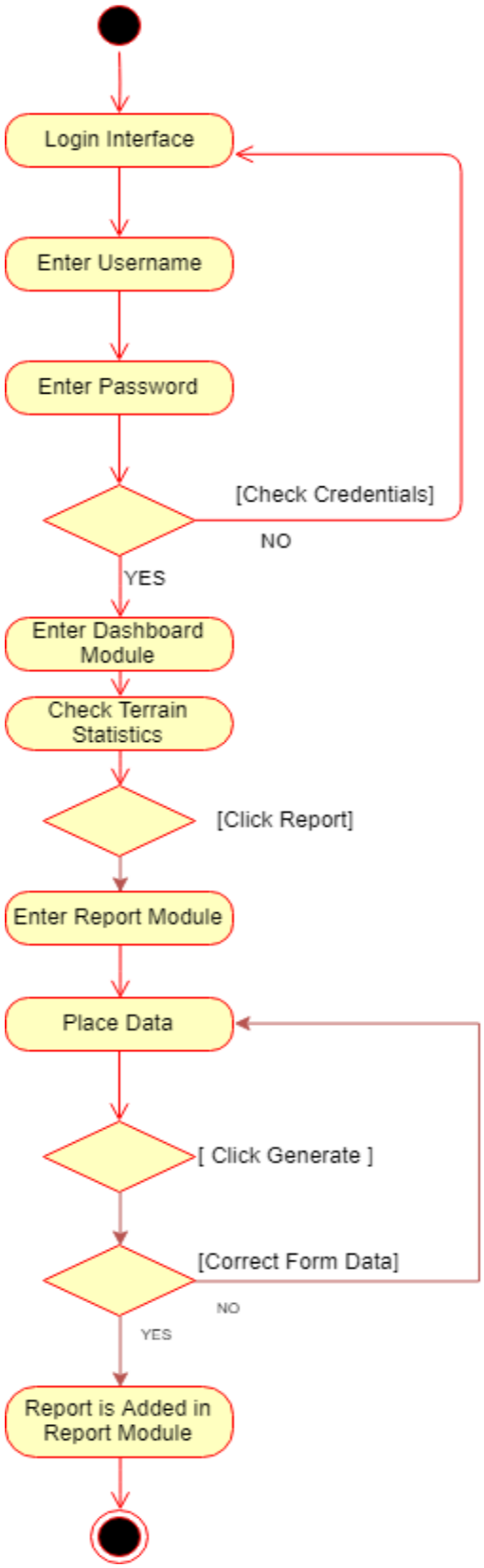
Human Resources Activity Diagram 3



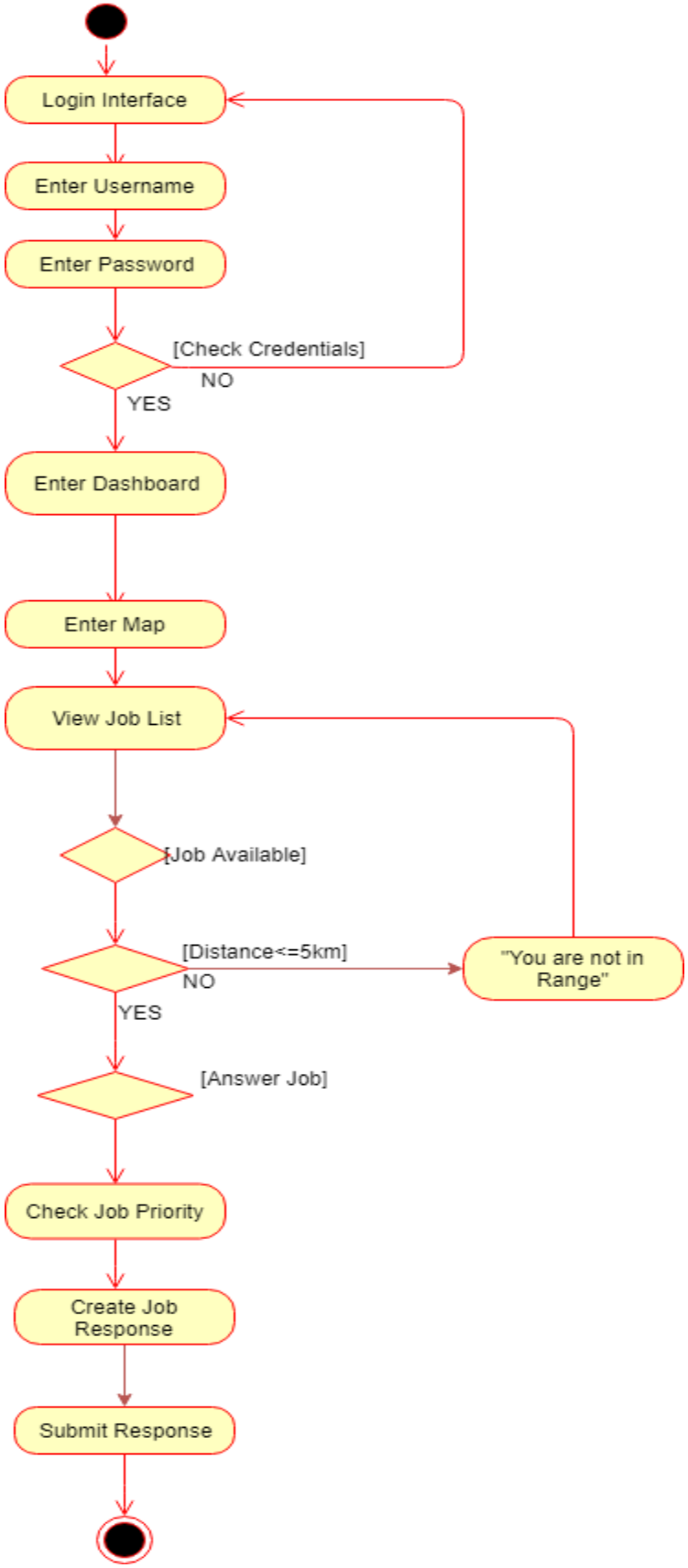
Human Resources Activity Diagram 4



Staff002 & Staff003 Activity Diagram

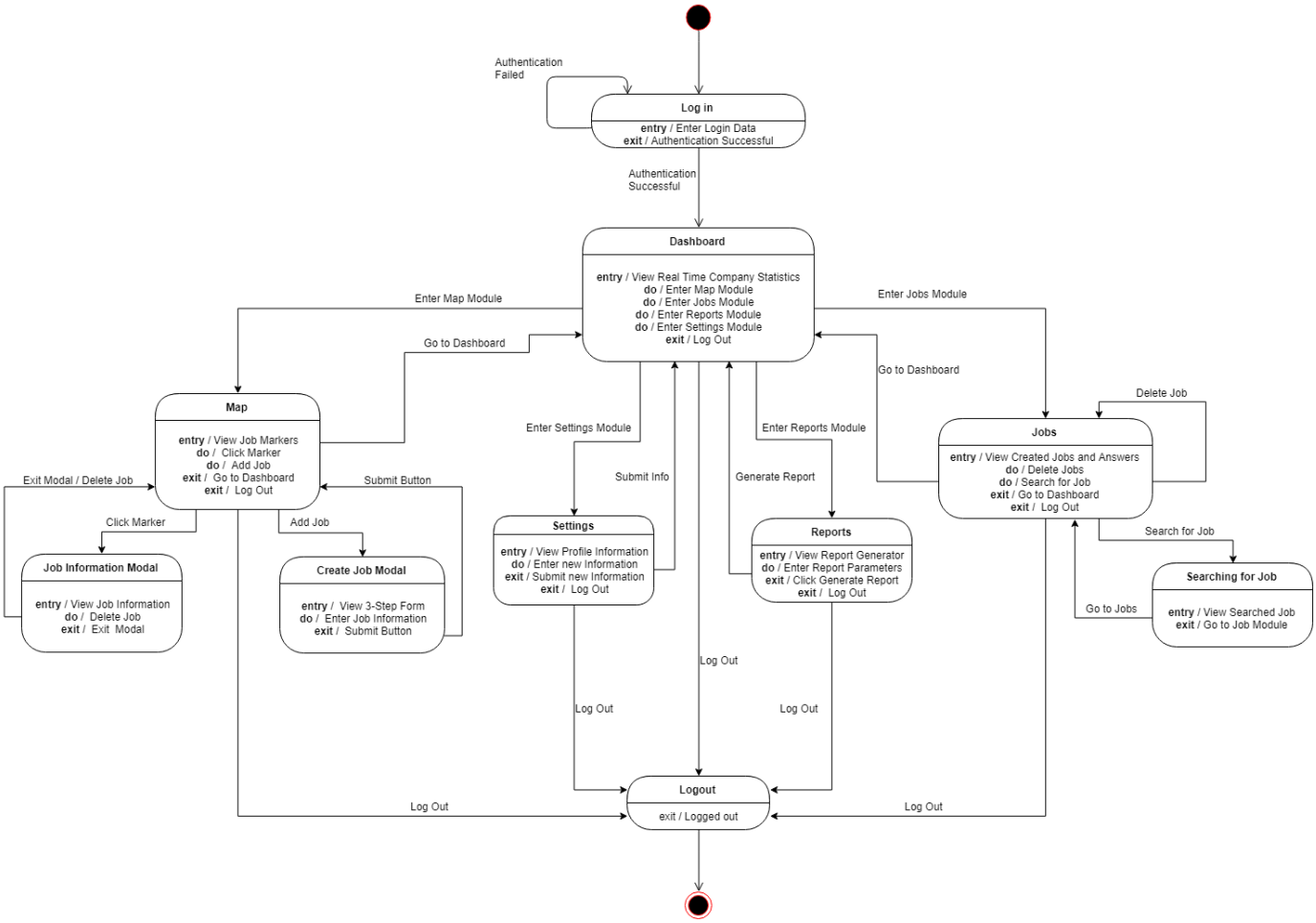


Terrain Inspector Activity Diagram

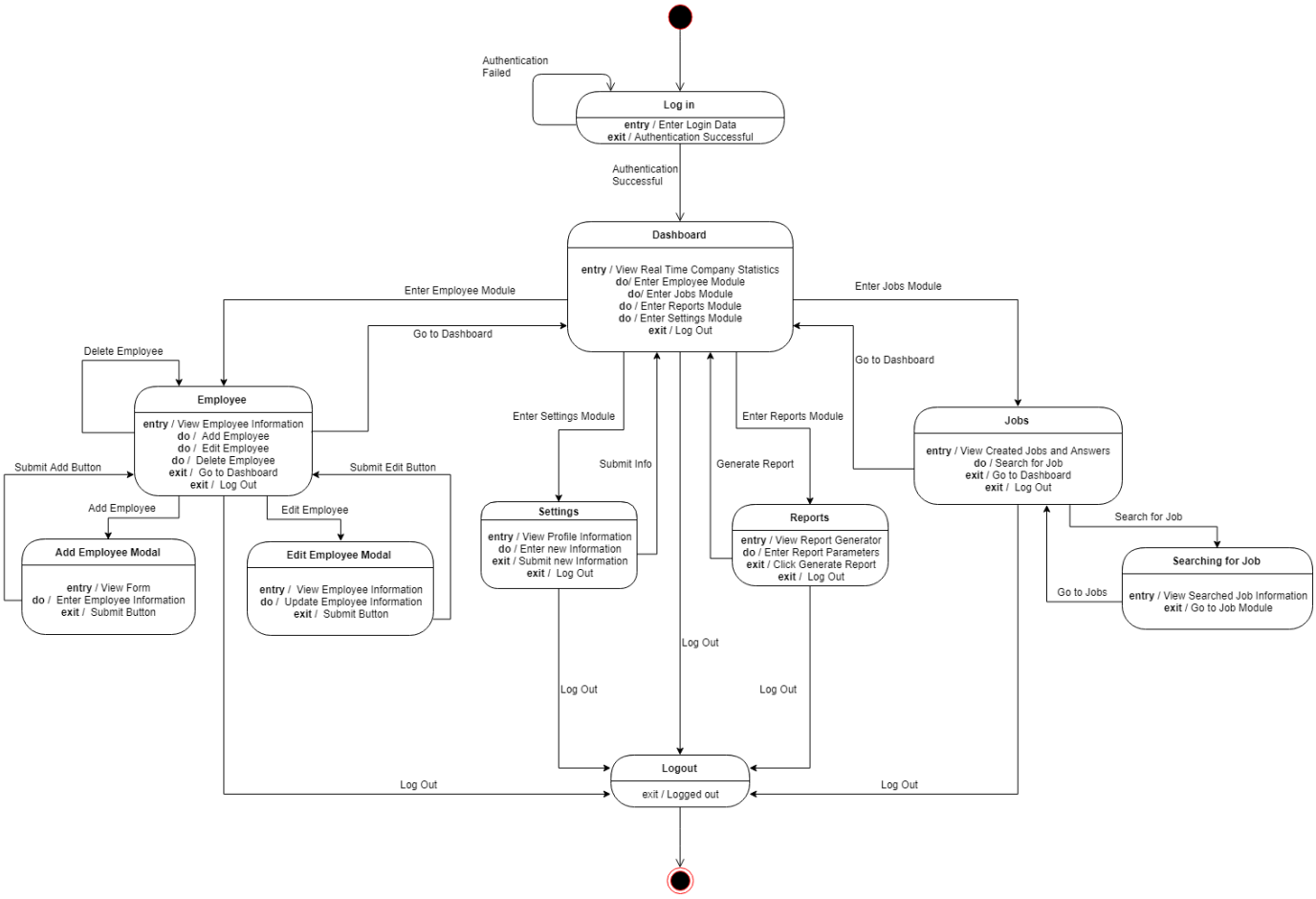


4.6 State Diagrams

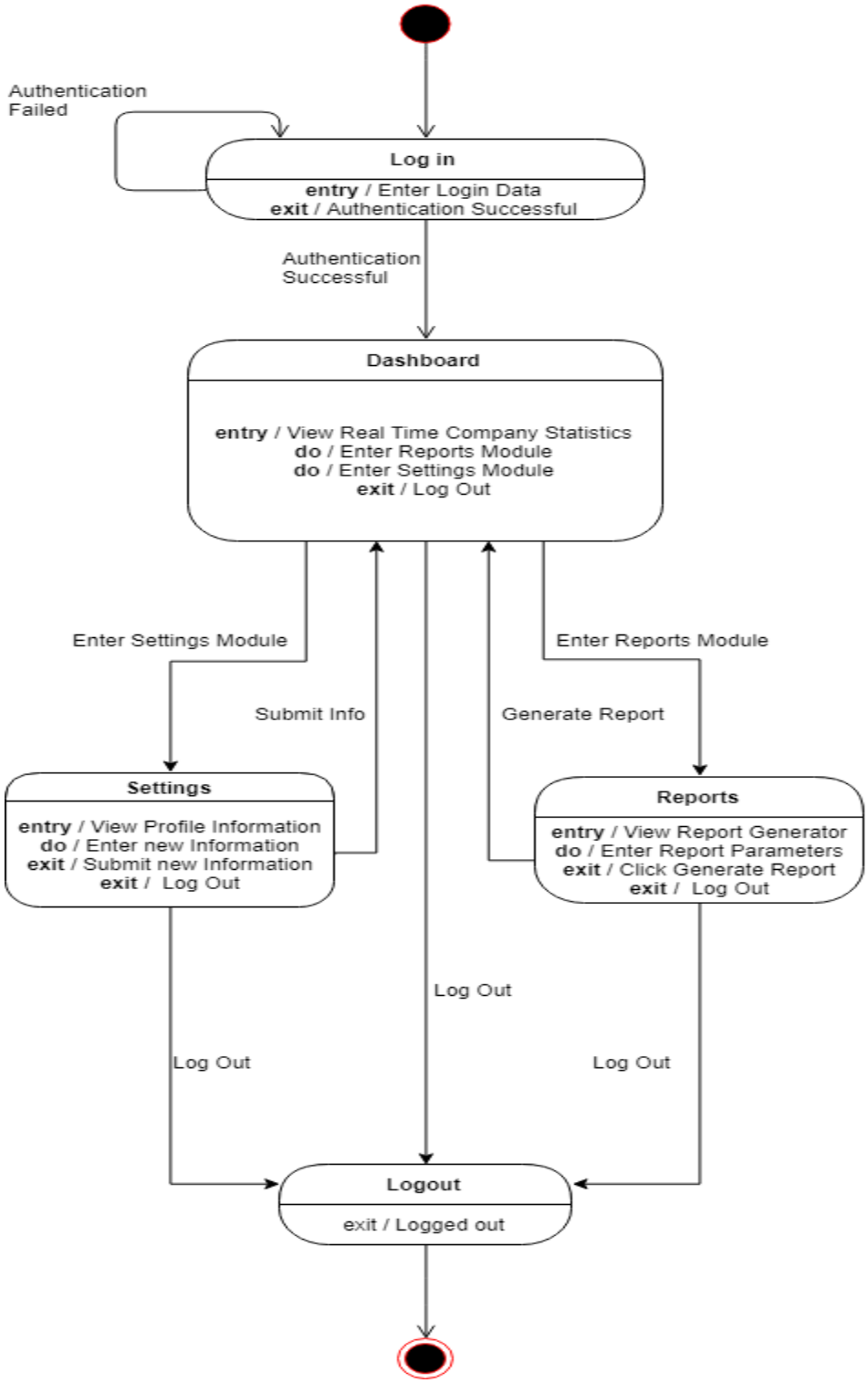
Chief Inspector State Diagram



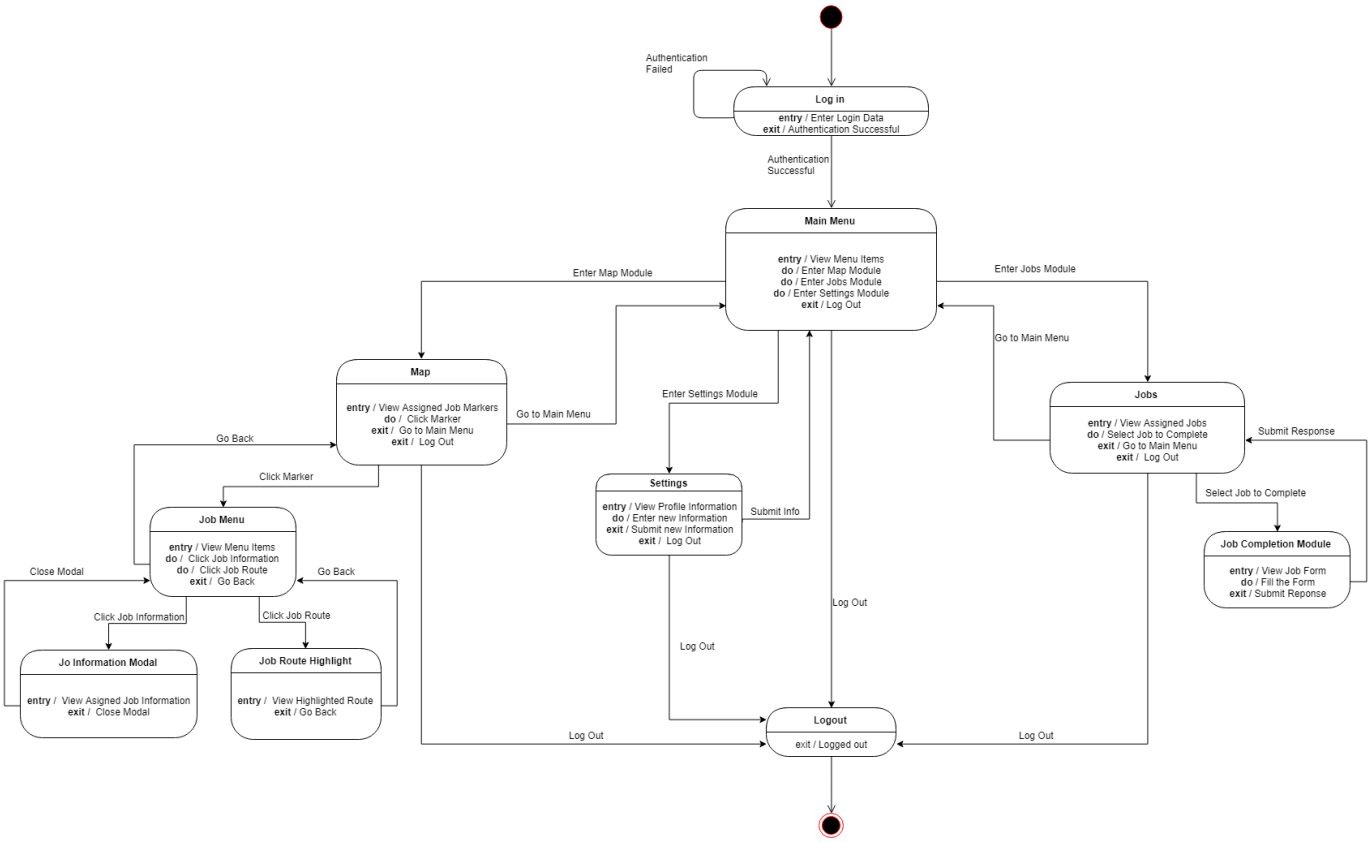
Human Resources State Diagram



Statistician and Economist State Diagram

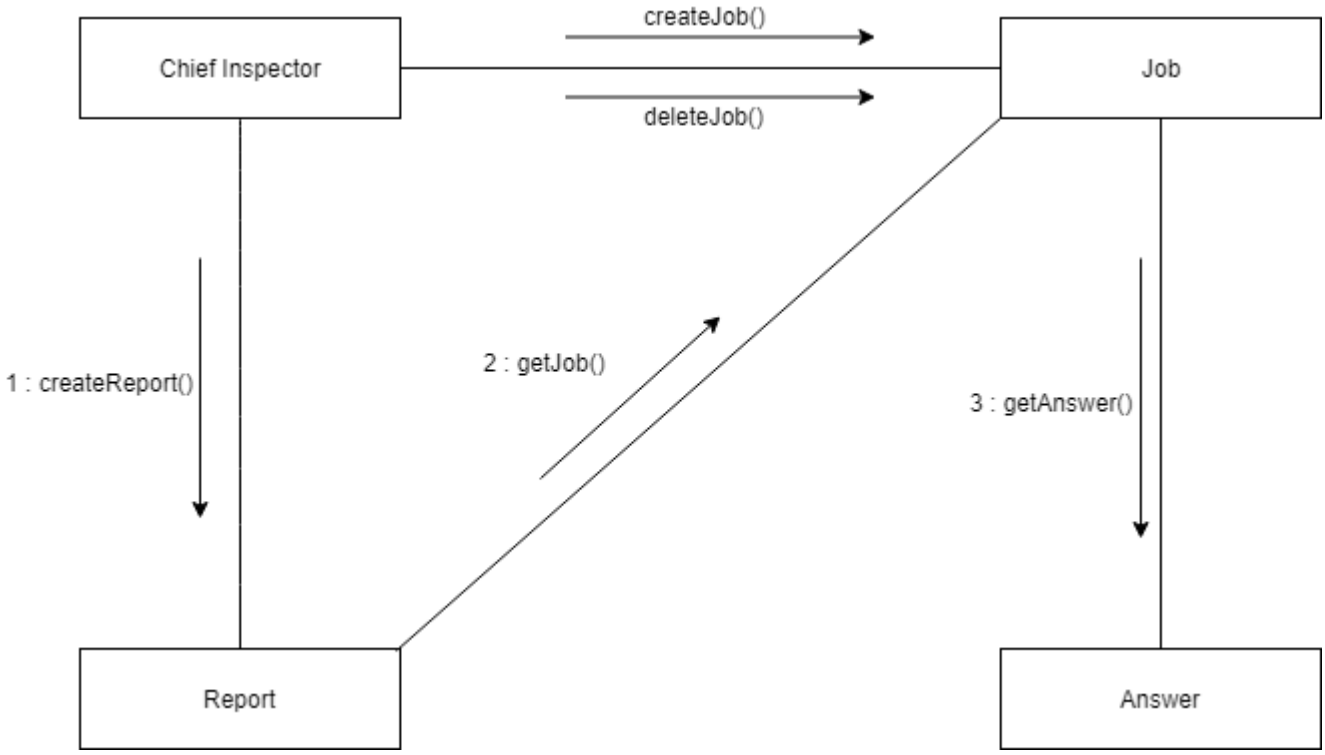


Terrain Inspector State Diagram

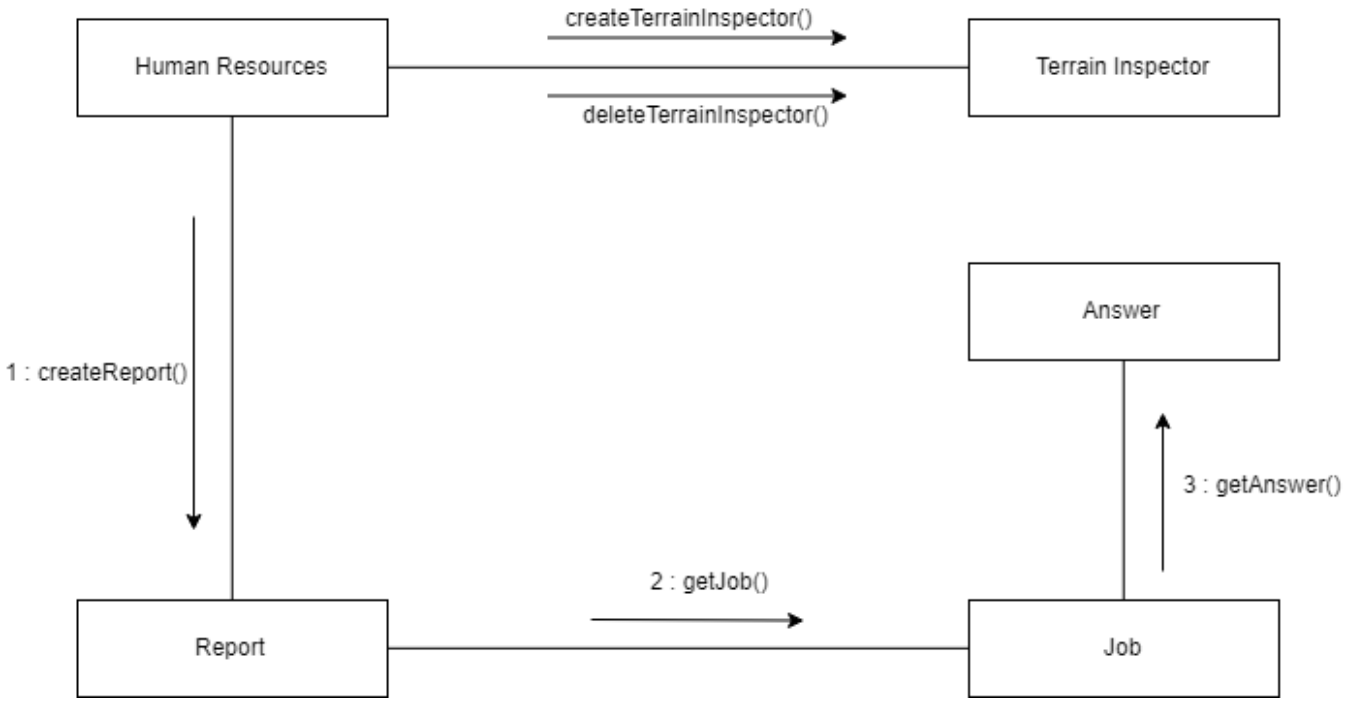


4.7 Collaboration Diagrams

Chief - Job - Answer - Report Collaboration Diagram

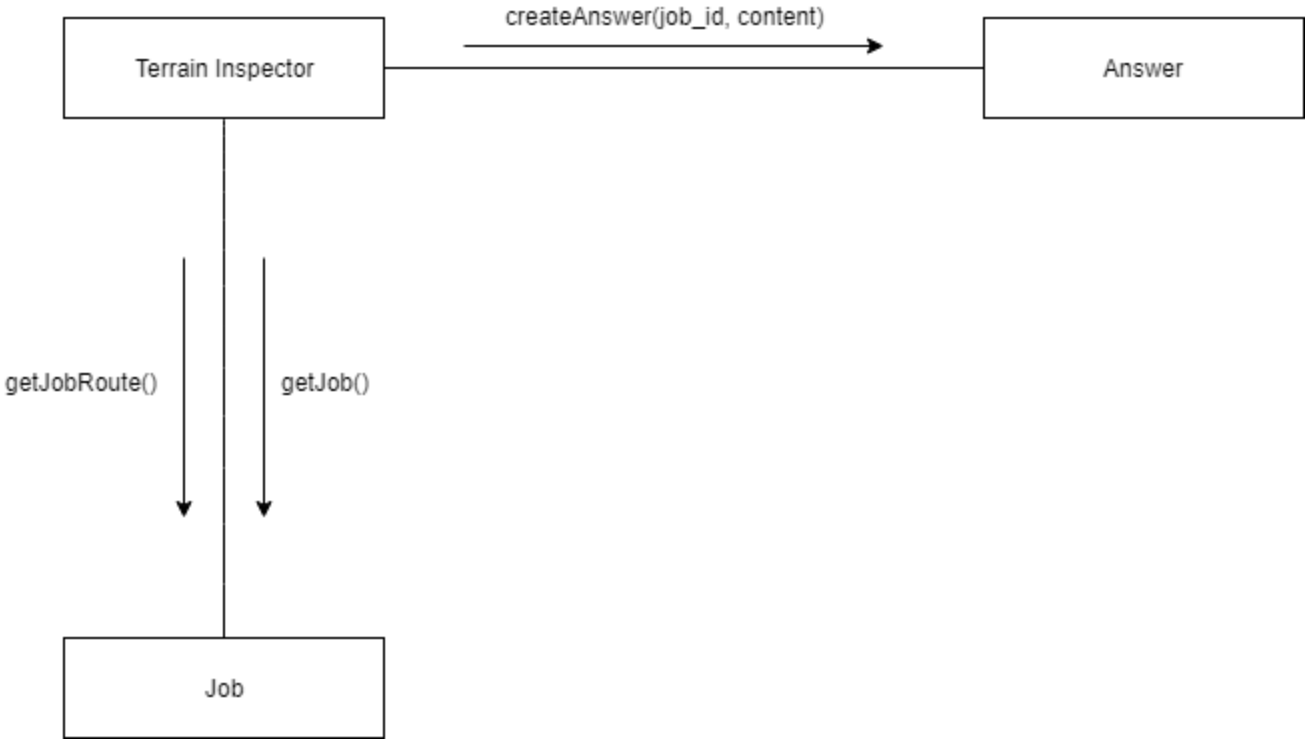


HR - Terrain - Report - Job - Answer Collaboration Diagram



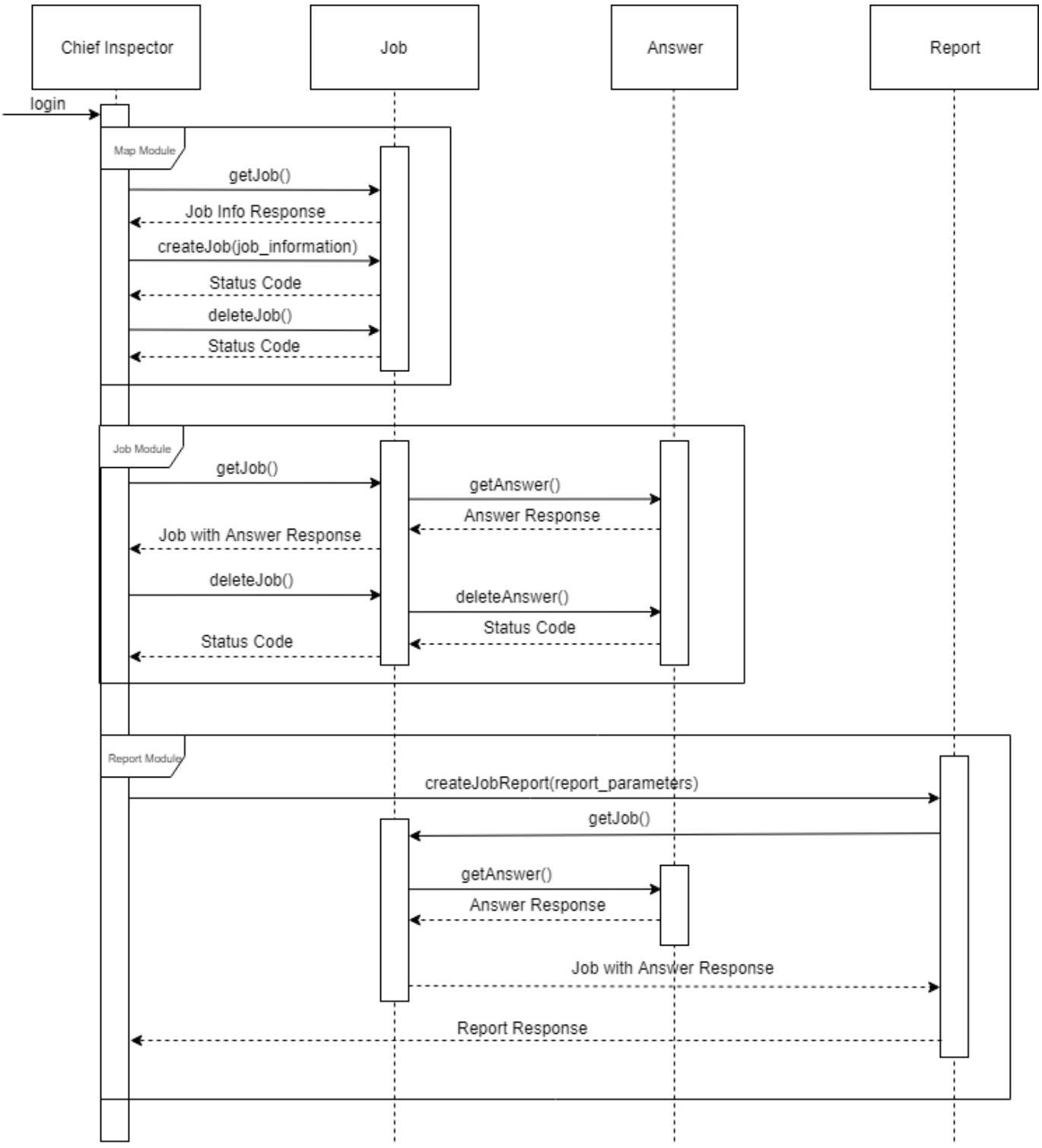
Staff - Job - Answer - Report Collaboration Diagram

Terrain - Job - Answer Collaboration Diagram

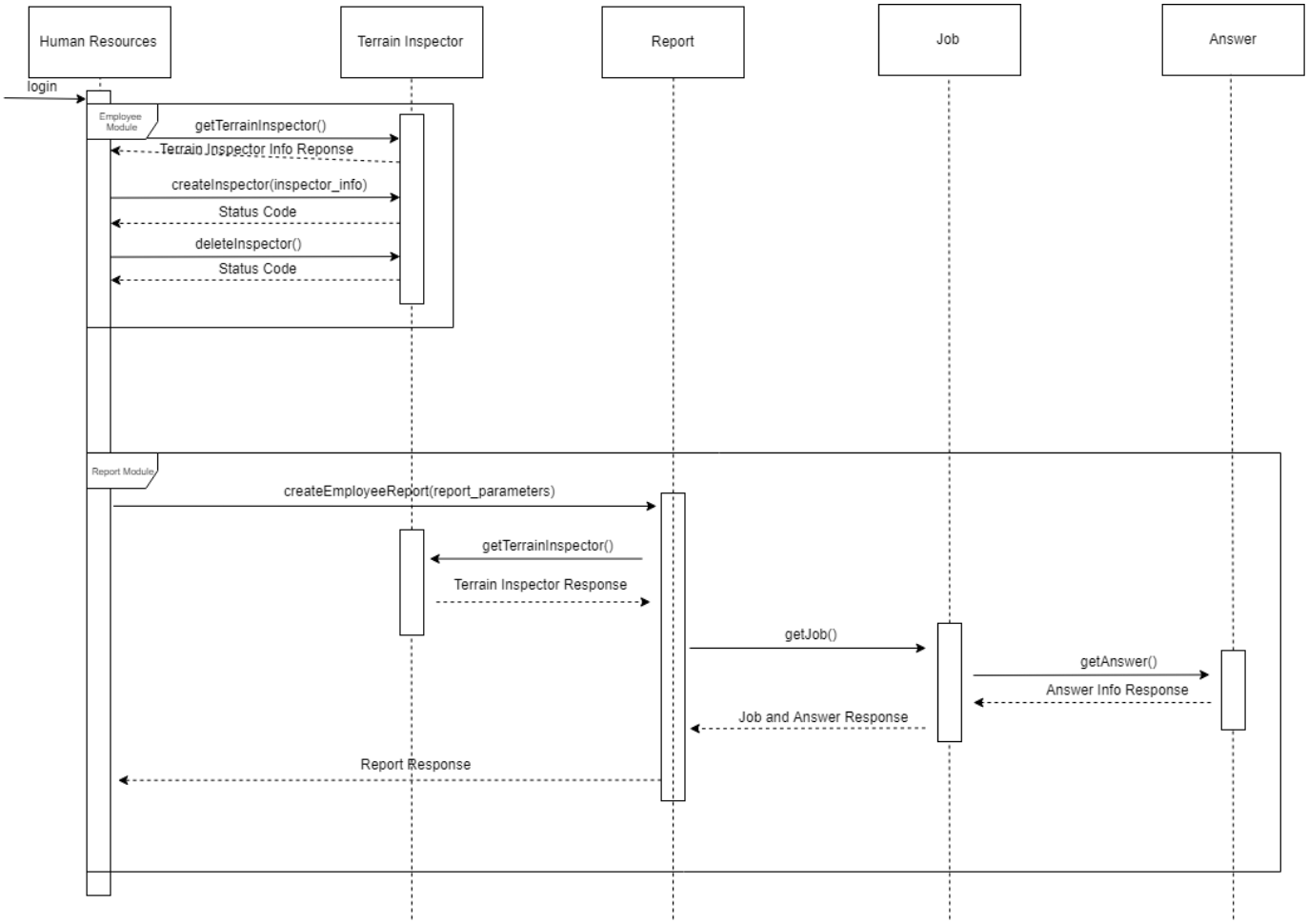


4.8 Sequence Diagrams

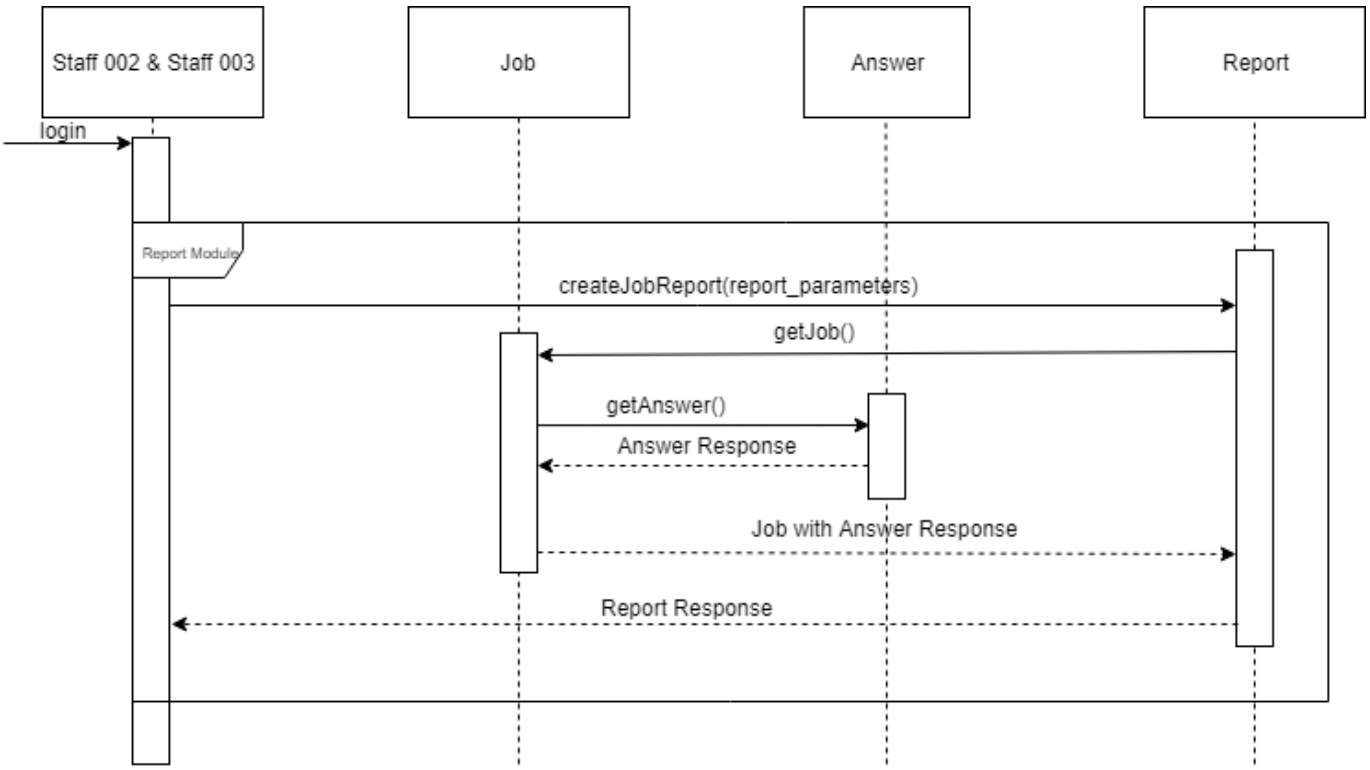
Chief Inspector Sequence Diagram



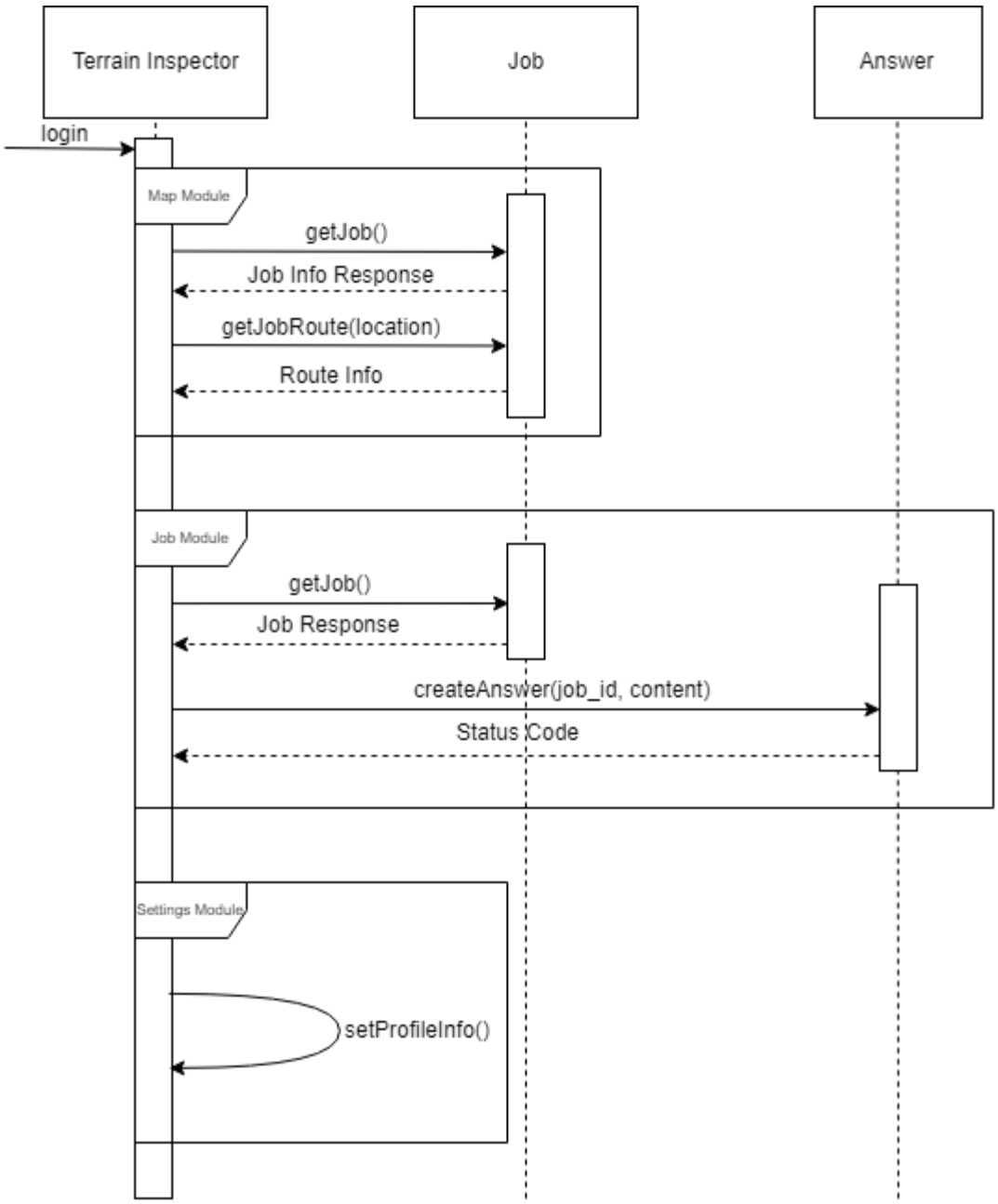
Human Resources Sequence Diagram



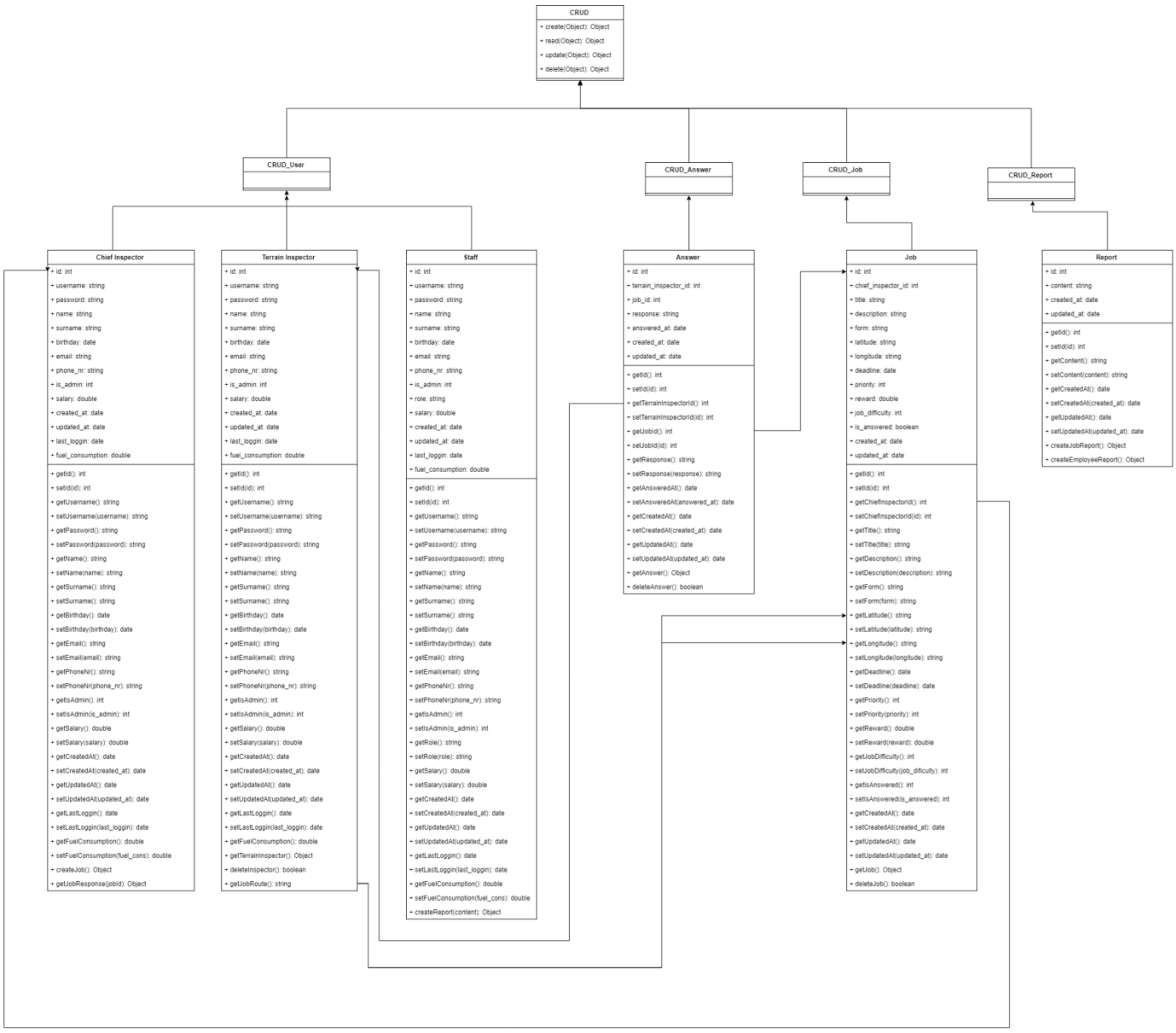
Staff002 & Staff003 Sequence Diagram



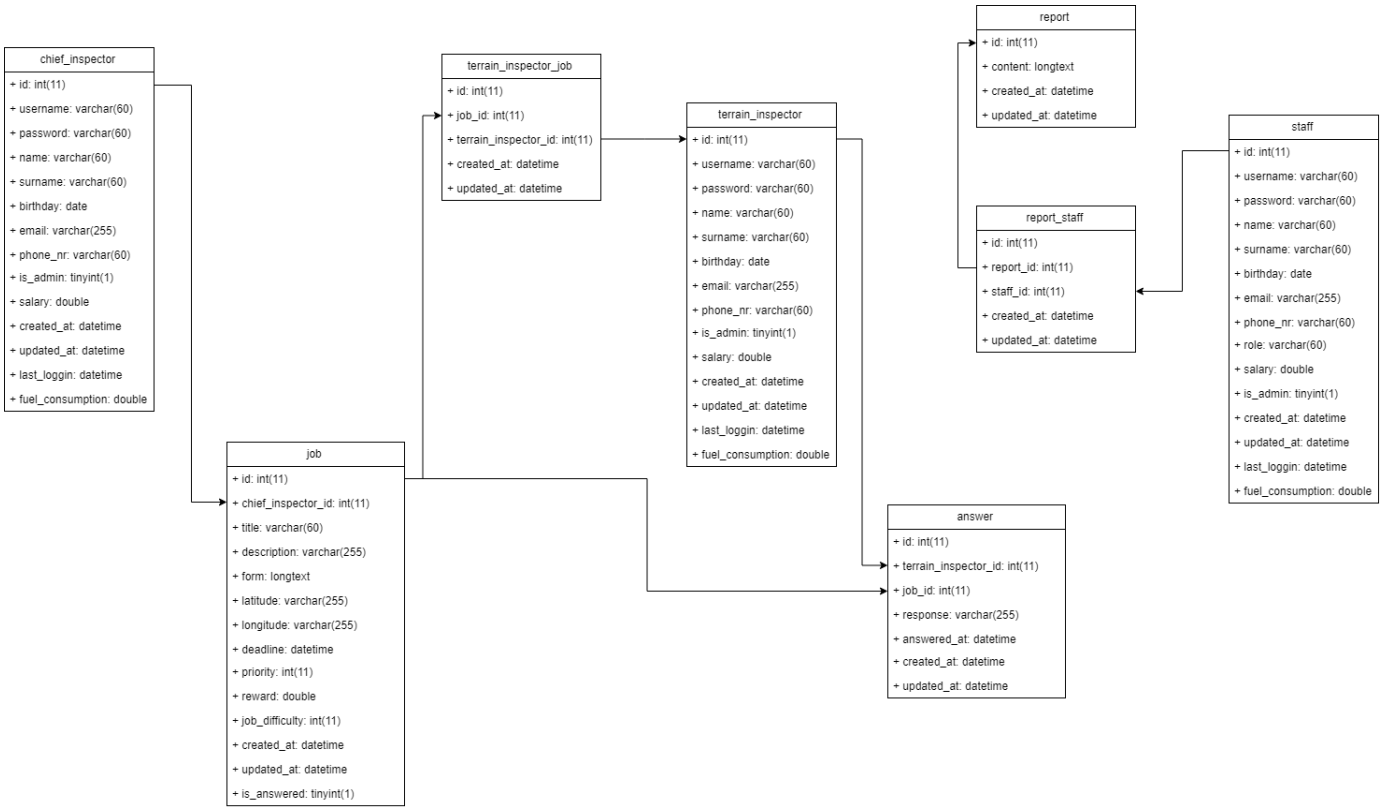
Terrain Inspector Sequence Diagram



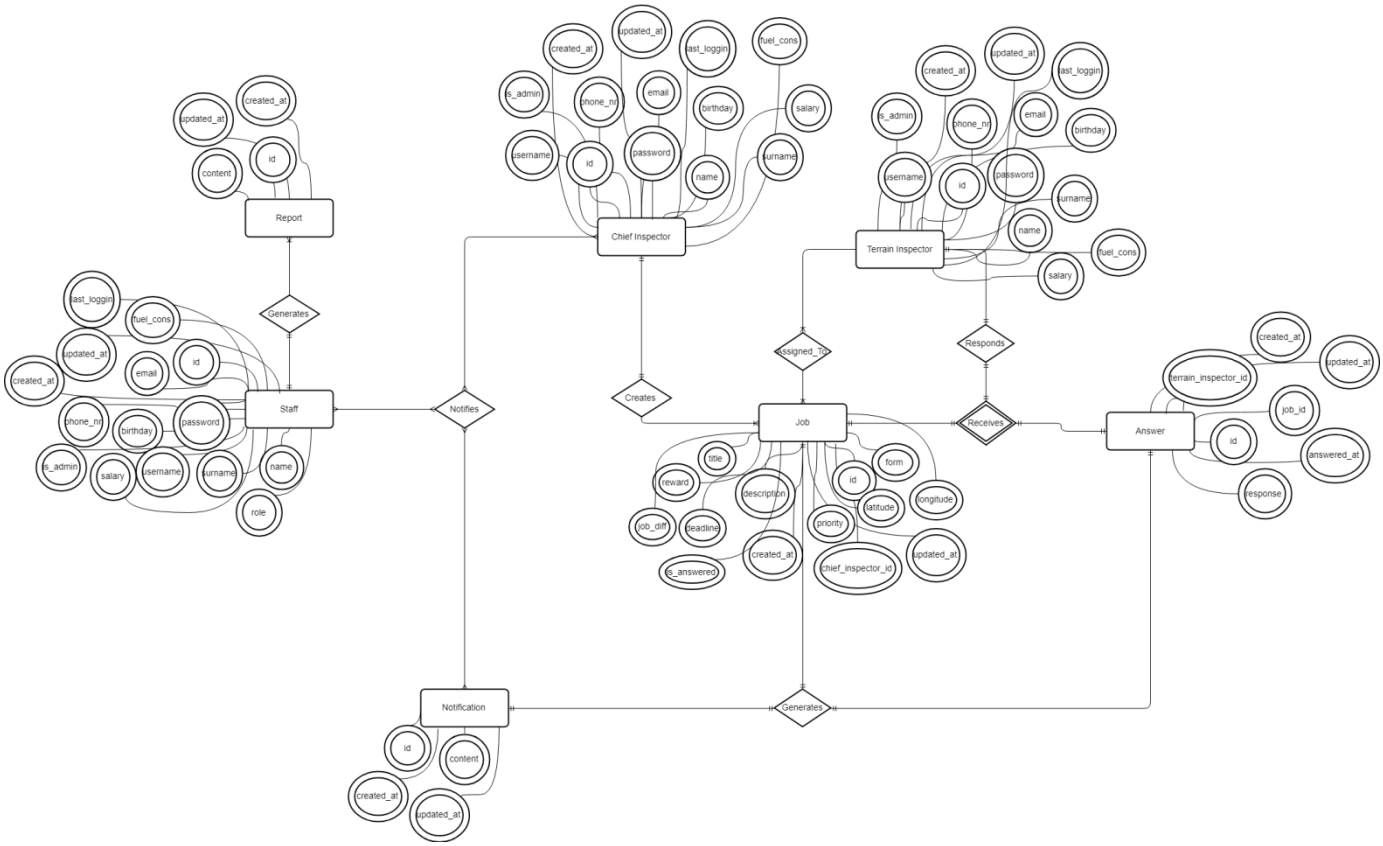
4.9 Class Diagram



4.10 DB Schema Diagram

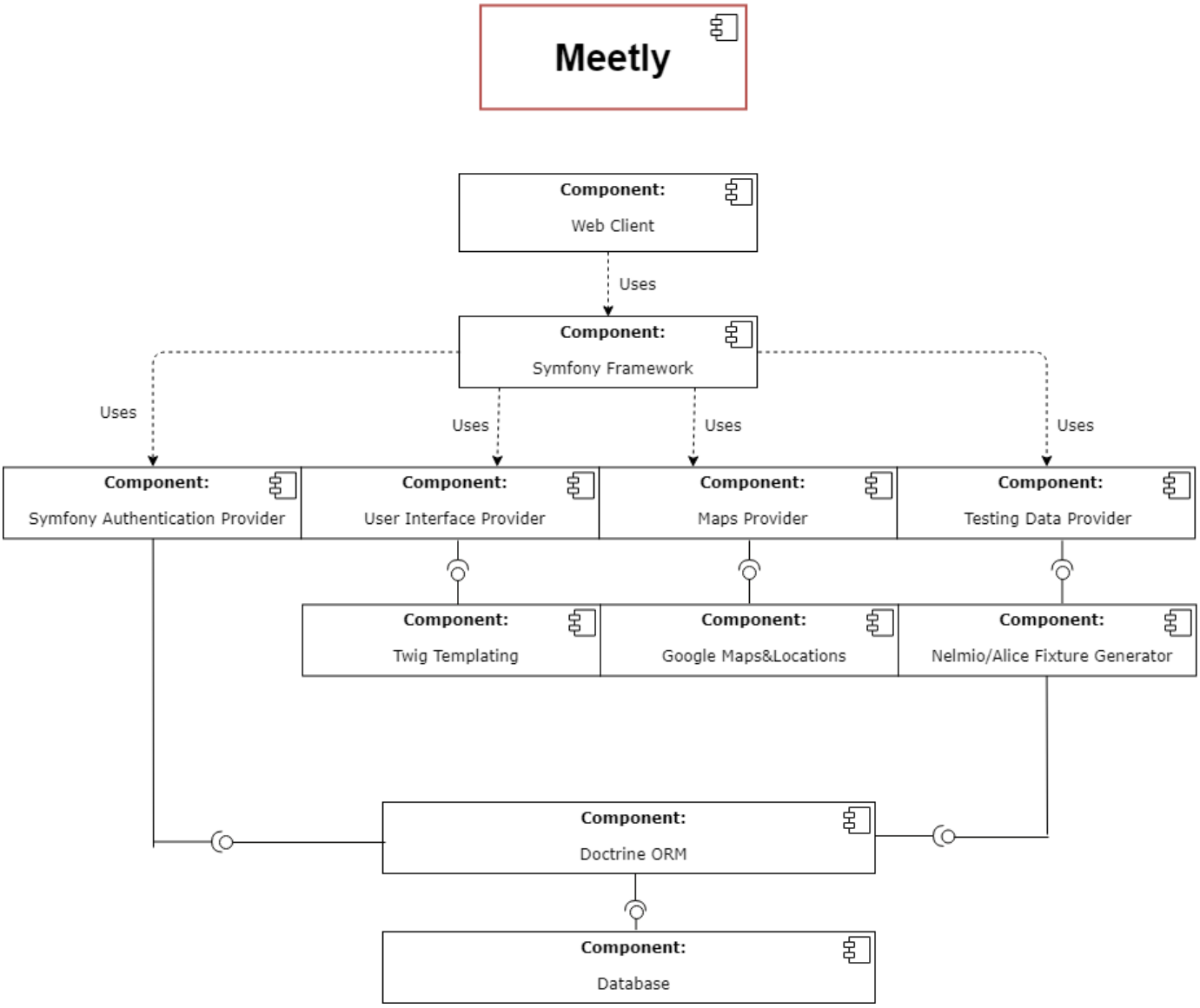


4.11 ER Diagram

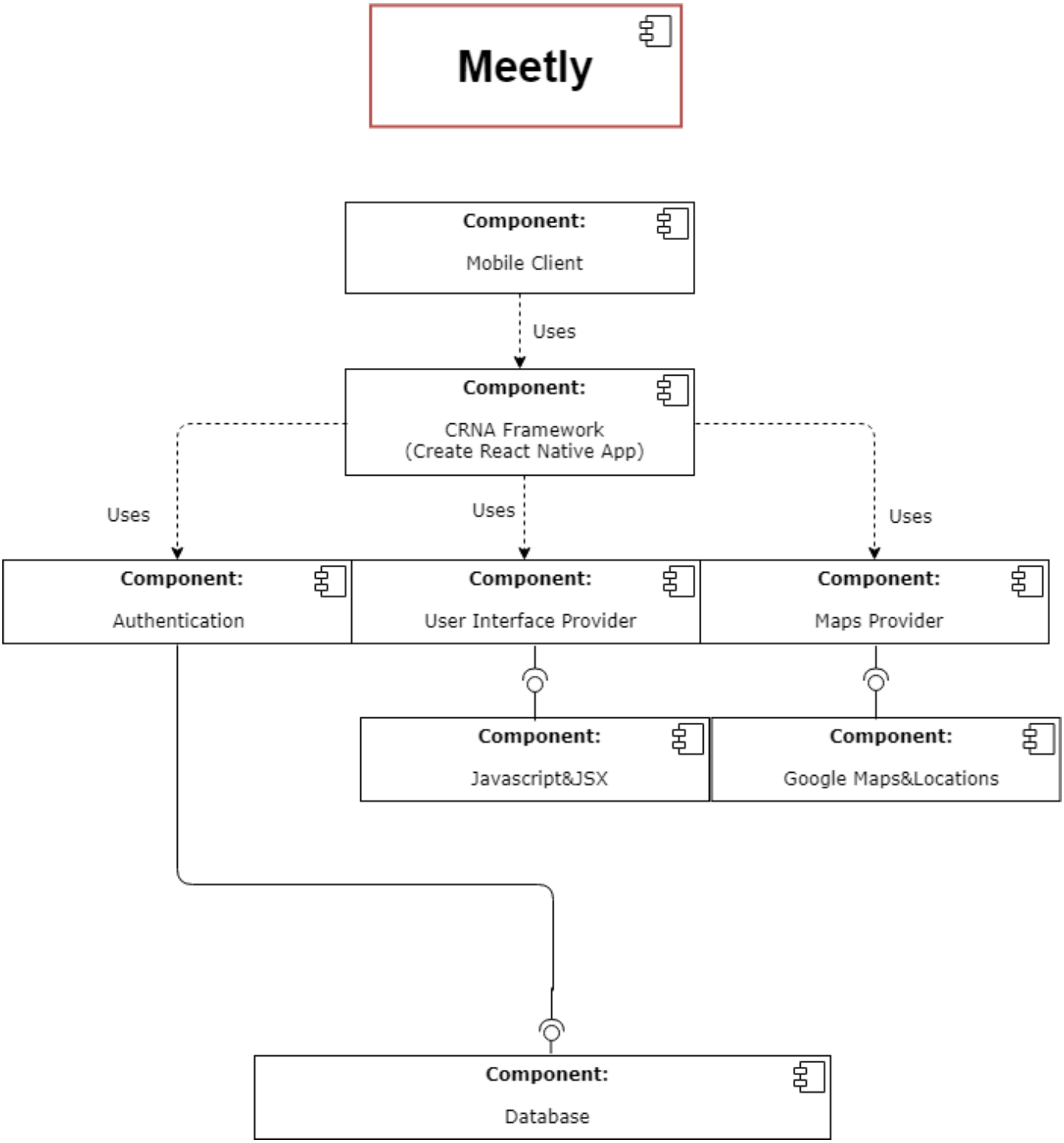


4.12 Component Diagram

Web Client Component Diagram

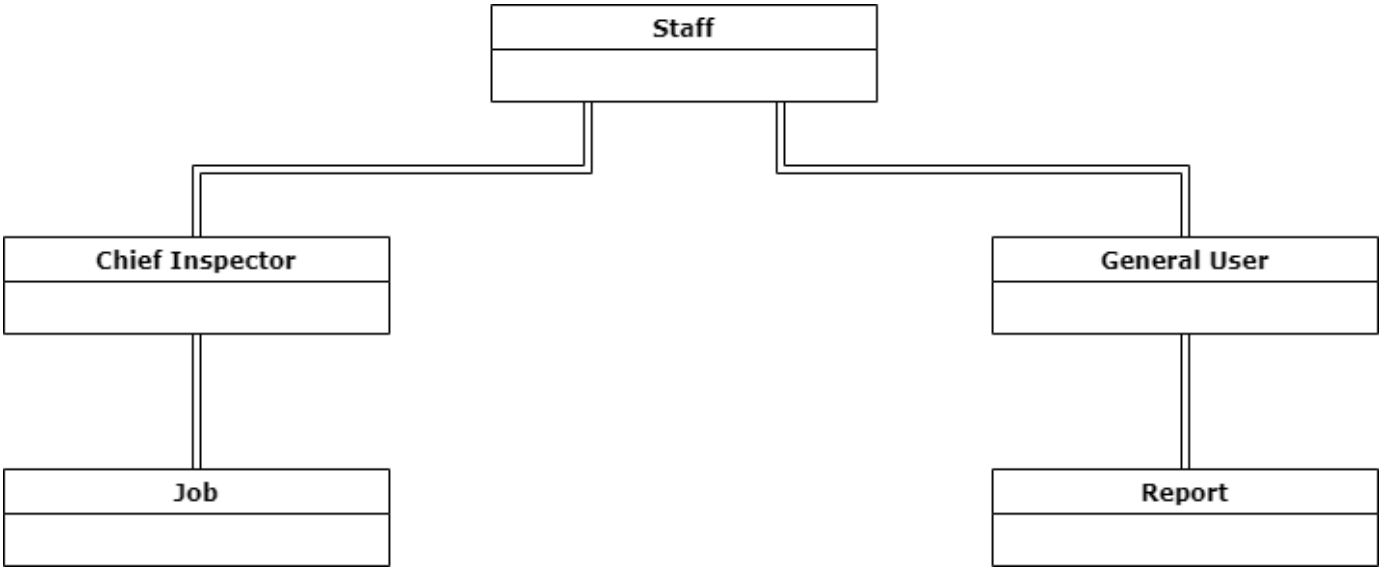


Web Client Component Diagram

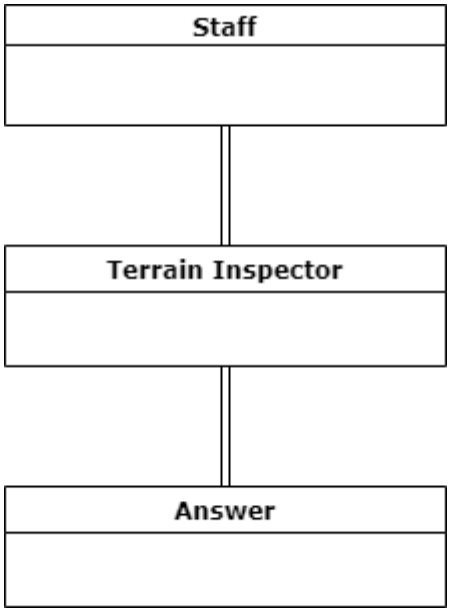


4.13 Object Diagram

Web Application Object Diagram

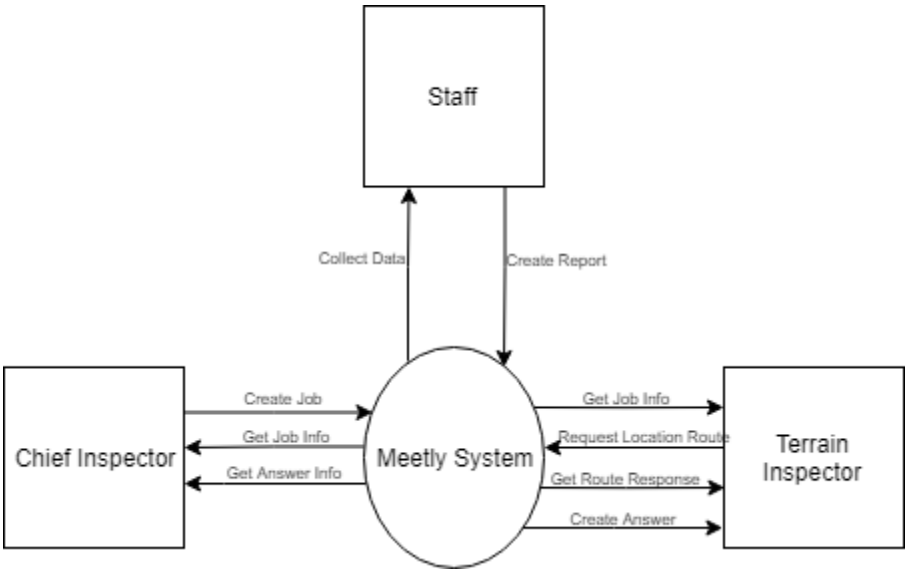


Mobile Application Object Diagram

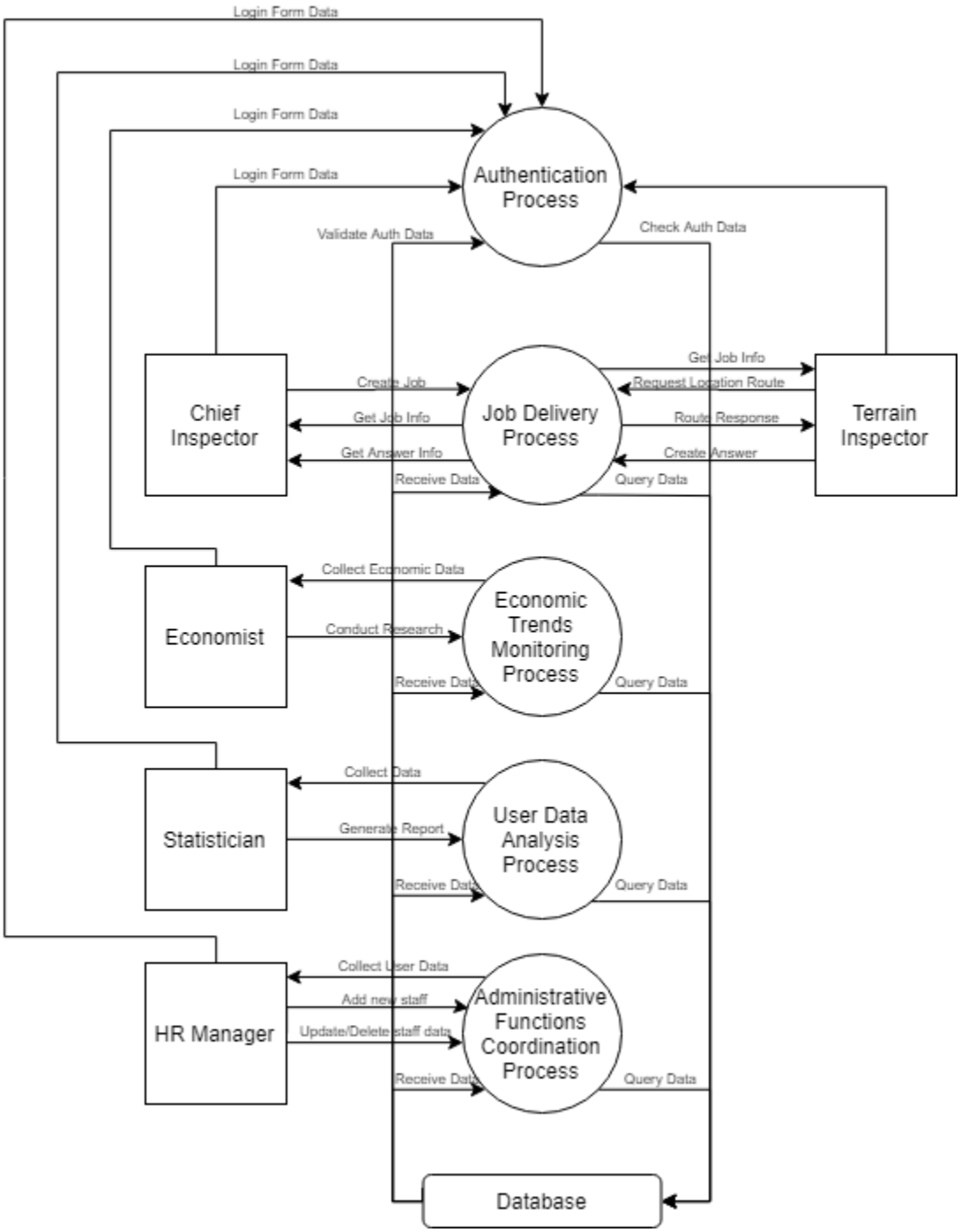


4.14 Data Flow Diagram

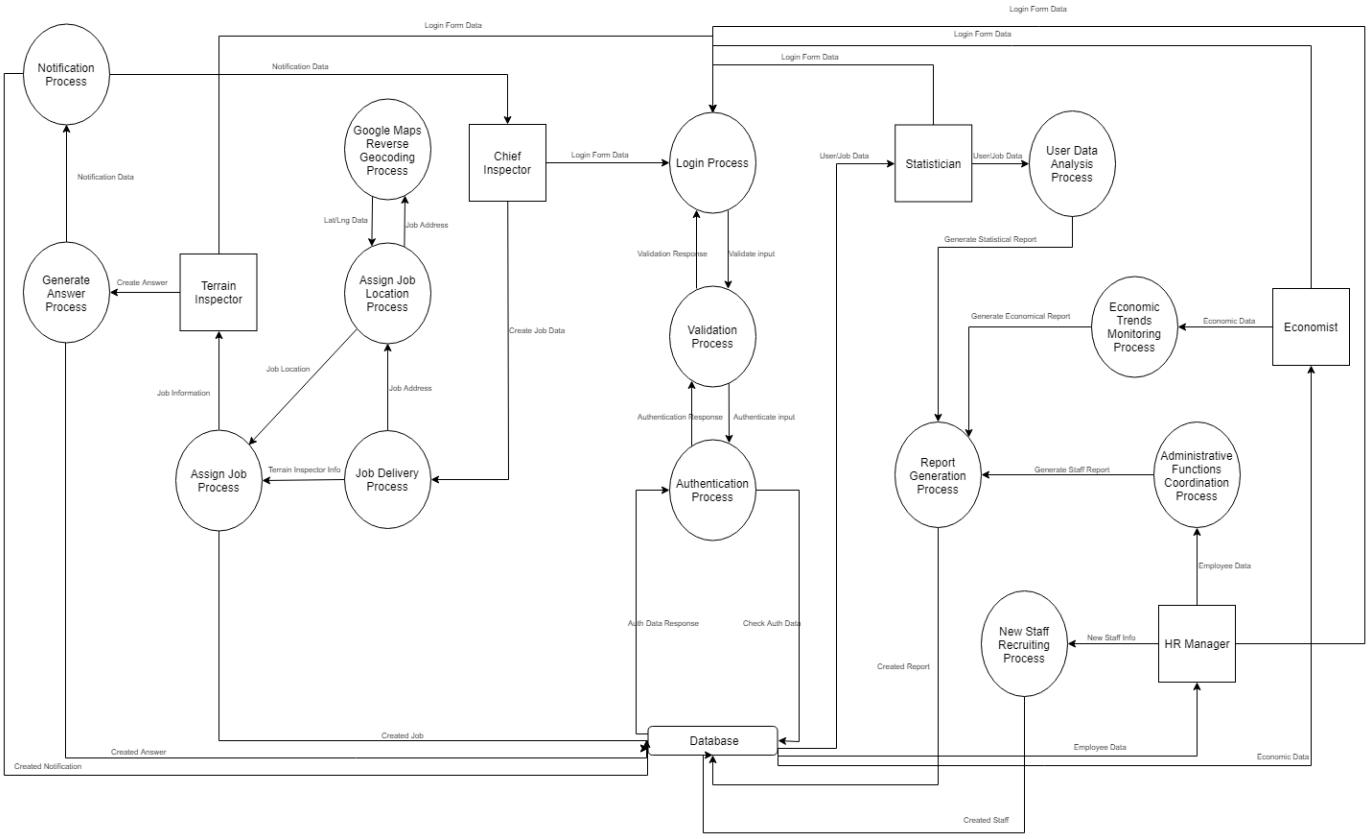
Level 0 DFD



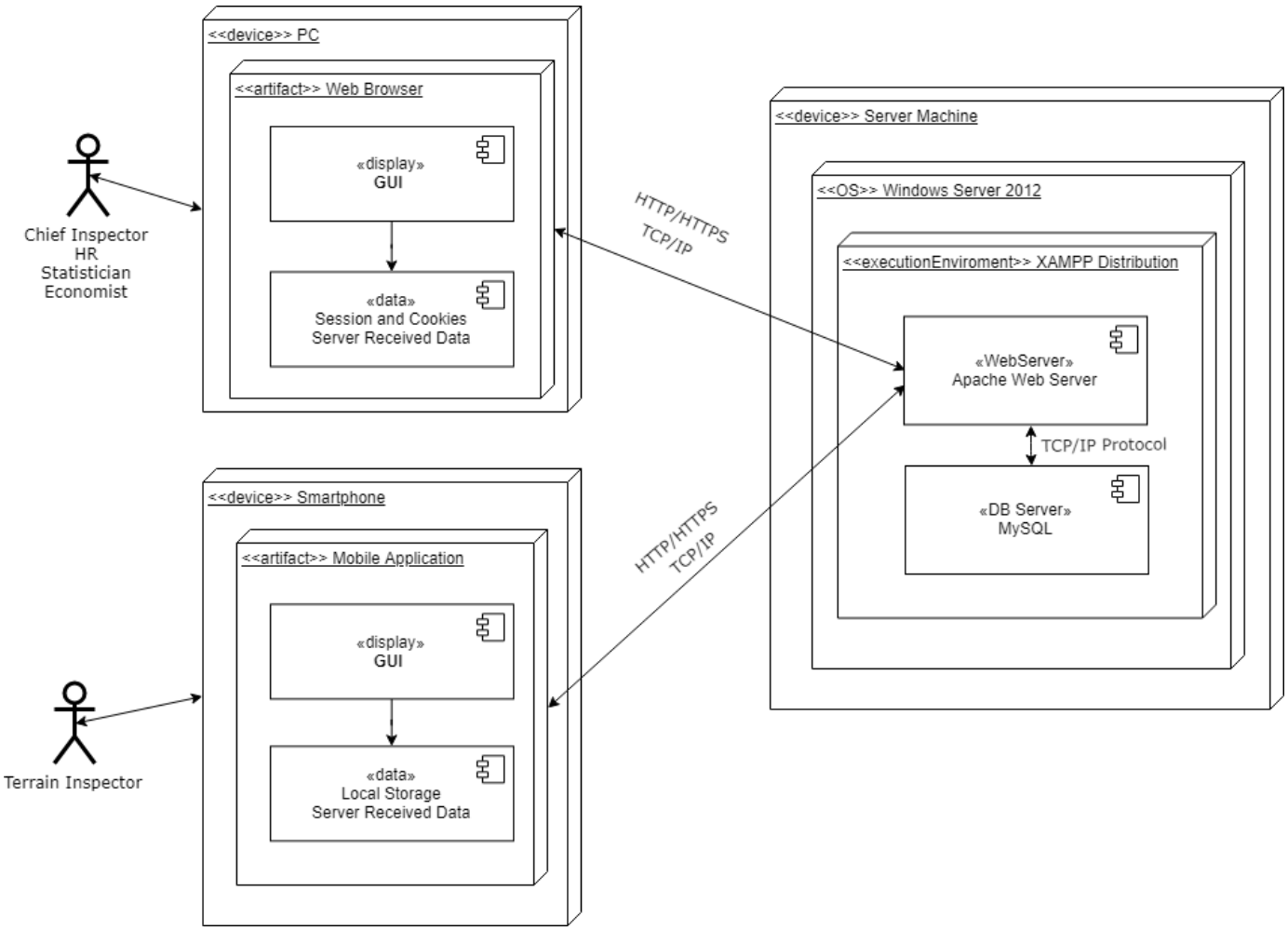
Level 1 DFD


Login Form Data
Login Form Data
Login Form Data
Login Form Data
Authentication Process
Validate Auth Data
Check Auth Data
Chief Inspector
Create Job
Get Job Info
Get Answer Info
Job Delivery Process
Receive Data
Query Data
Get Job Info
Request Location Route
Route Response
Create Answer
Terrain Inspector
Economist
Collect Economic Data
Conduct Research
Economic Trends Monitoring Process
Receive Data
Query Data
Statistician
Collect Data
Generate Report
User Data Analysis Process
Receive Data
Query Data
HR Manager
Collect User Data
Add new staff
Update/Delete staff data
Administrative Functions Coordination Process
Receive Data
Query Data
Database

Data Flow Diagram - Level 2



4.15 Deployment Diagram



5. Implementation Technology

To implement this project we decided to use a combination of new and innovative technologies, while maintaining the standardized methods of Software Engineering.

Meetly is a combination of paradigms involving a web application and a mobile application, where every part of the system serves for a specific purpose. To describe the technology used let's first explain the Web Application and then we continue to the Mobile Application.

Meetly Web App – The Meetly web application is implemented in the Symfony framework for web development. The choice behind Symfony is mainly concentrated in the following points:

1. Permanence of the framework and its long term support.

2. The framework promotes and enforces best practices, standardization and interoperability of applications.

3. The framework enforces a MVC model.

4. The framework makes testing and debugging very easy and comprehensive.

5. The framework supports very well mid to large scale web applications, which is the type of project that we aim to build.

In order to use this framework efficiently, we were required to use the following languages/markups:

Frontend Development: HTML/Twig Templating, CSS, Javascript

Backend Development: PHP

Configuration: XML

Database: Doctrine ORM/ MySQL

Meetly Mobile App – The Meetly mobile application is implemented in the React Native framework for cross-platform mobiledevelopment. The choice behind R is mainly concentrated in the following points:

1. The framework is tried and tested. Facebook built React Native first and foremost to create a fantastic mobile app for their own social portal. Since then popular apps like Facebook, Instagram, Airbnb, Skype, Tesla, Walmart etc.. have used React Native for their mobile application.

2. Support of one codebase for two platforms, IOS and Android.

3. Fast, since it compiles to native Java and Swift code.

4. Use of javascript for every process is an added benefit.

In order to use this framework efficiently, we were required to use the following languages/markups:

Frontend Development: Javascript(ES6), JSX

Backend Scripts: PHP and MySQL

In order to use the technologies mentioned above we have also used the following programming paradigms:

Object Oriented Programming – For every type of user, job, answer etc.. we were required to build Classes based on OOP rules.

Asynchronous Calls or AJAX – Each time we had to interact with the database without changing our web/mobile view, we used AJAX calls through javascript to our server.

Database Triggers – In order to trigger notifications on any new job or answer, we had to use database triggers in MySQL.

User Interface Details:

For the user interface design of the web application, we used Bootstrap classes with a number of self-created styles.

For the user interface design of the mobile application, we designed and coded every style used.

Hosting:

This whole project will be hosted in a personal server and also in the stud-proj.epoka.edu.al, together with the MySQL database.

Networking:

For networking, this project will take advantage of the HTTP protocol.

6. Project Planning

Project Name: Meetly

Members: Deni Daja, Klesti Kuka, Bojken Sina, Ernit Hoxha

Start Date: 15.03.2018

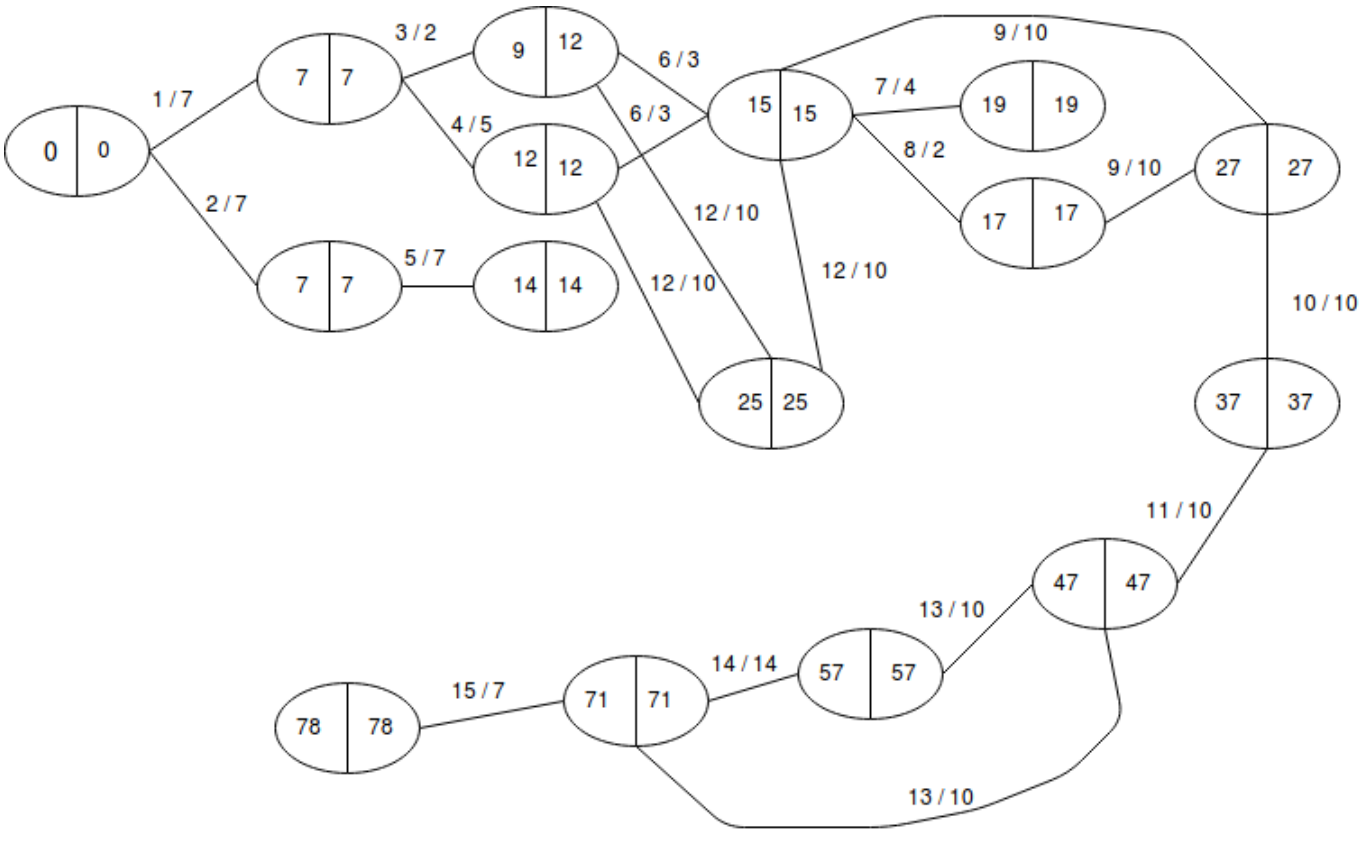
End Date: 28.09.2017

Duration: 194 Days

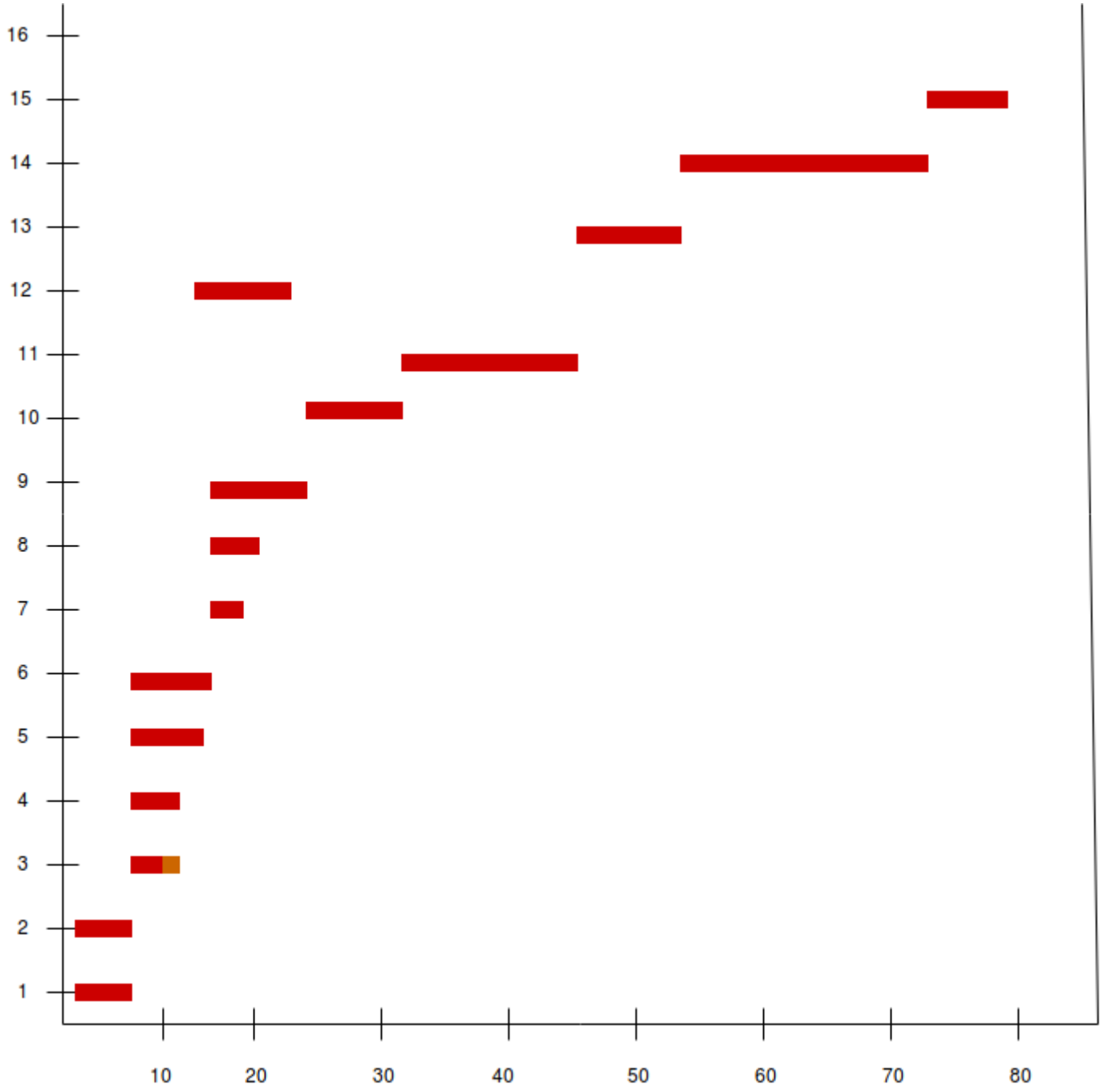
Nr	NAME	DURATION	DEPENDENCY
1	Decide On Project Name and Type	7 days	
2	Decide On Projects Technology	7 days	
3	Write Project Description	2 days	1
4	Determine Functional Requirements	5 days	1

5	Determine Non-Functional Requirements	7 days	2
6	Determine User Scenarios	3 days	3,4
7	Determine User Cases	4 days	6
8	Write User Case Diagrams	2 days	6
9	Write Behavioral Diagrams	10 days	8,6
10	Write DFD Diagram	10 days	9
11	Define Structural Diagrams	10 days	10
12	Draw Sketches	10 days	3,4,6
13	Define ERD and Table Structure	10 days	11
14	Code Generation	14 days	13,11
15	Define Software Project Management	7 days	14

NETWORKS



STAGE PLAN (GANTT CHARTS)



APPENDIX

APPENDIX A

Definitions, Abbreviations

Code Conventions

Staff001 - Human Resources (HR)

Staff002 - Economist

Staff003 - Statistician

Abbreviations :

CRUD – Create, Read, Update, Delete

HR – Human Resources

CI – Central Inspectorate

PHP – Hypertext Preprocessor

SQL – Standard Query Language

CRNA – Create React Native App

GPS **–** Global Positioning System

APPENDIX B

References

[1]. http://www.insq.gov.al/inspektorati-shteteror-i-punes-dhe-sherbimeve-sociale/

[2]. http://www.insq.gov.al/ligje/

[3]. https://www.creative-tim.com/product/paper-dashboard

[4]. Paper Dashboard: [http://www.comentum.com/guide-to- web-applicationdevelopment.html]

[5]. https://symfony.com/doc/current/index.html

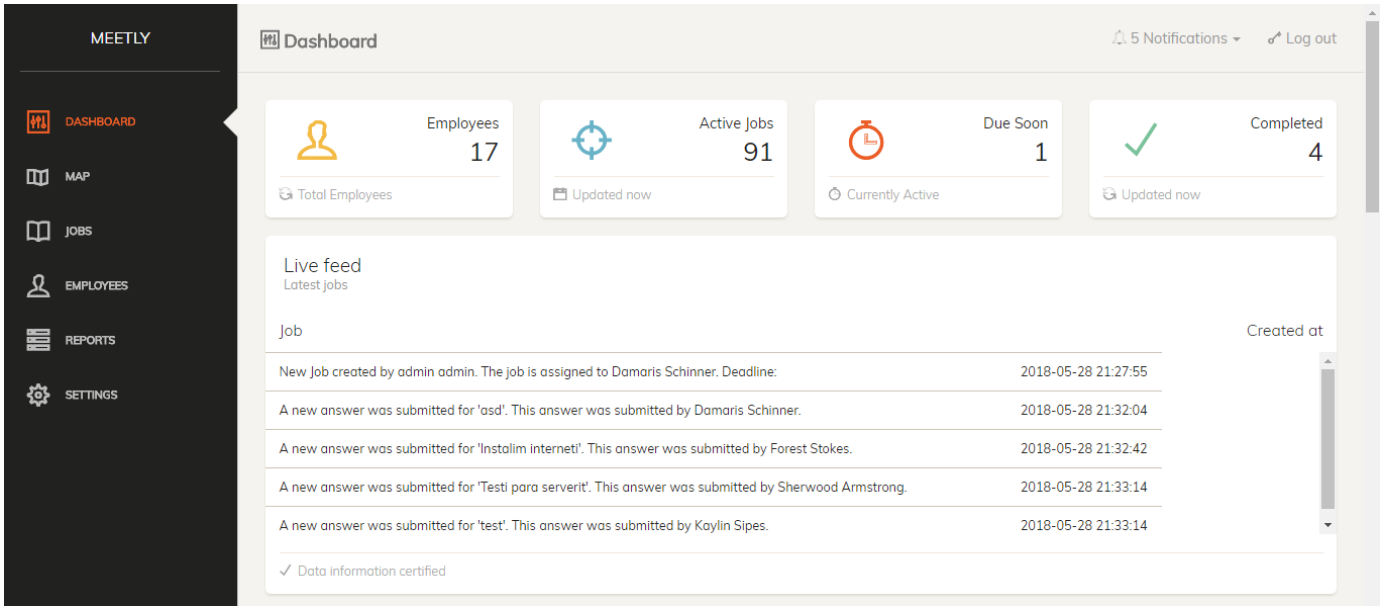
[6]. https://facebook.github.io/react-native/docs/getting-started.html

[6]. Server where the web app resides: [http://stud-proj.epoka.edu.al/~ddaja15/meetly/]

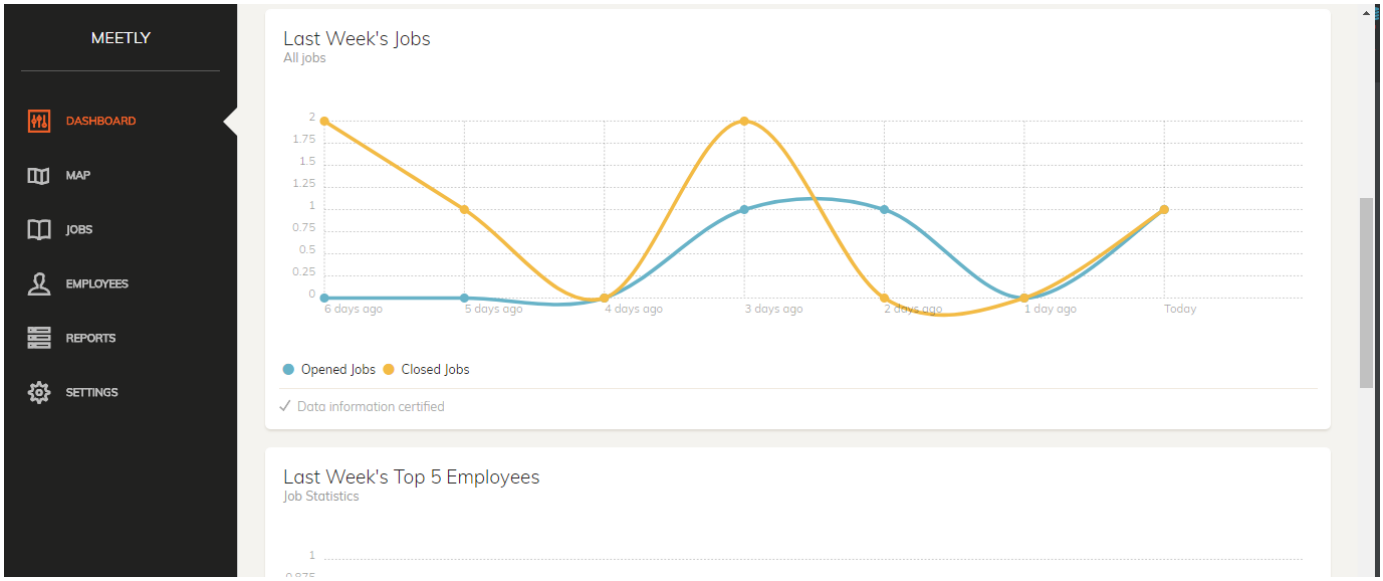
APPENDIX C

Screenshots

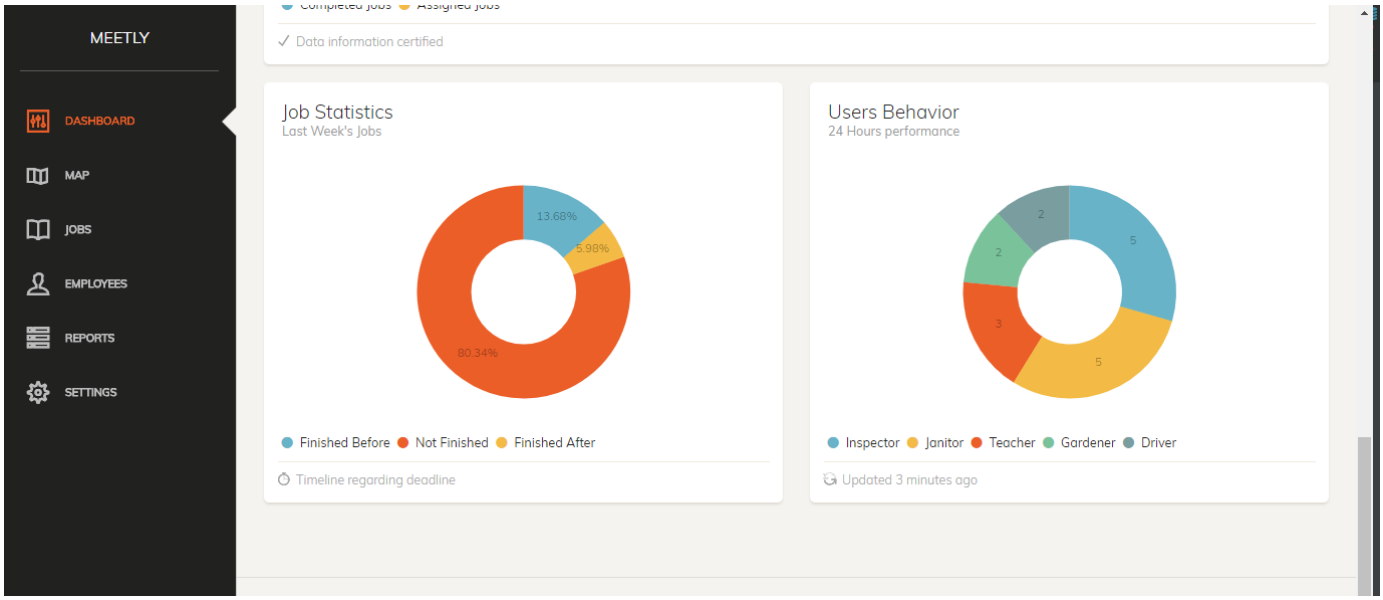
Dashboard 1



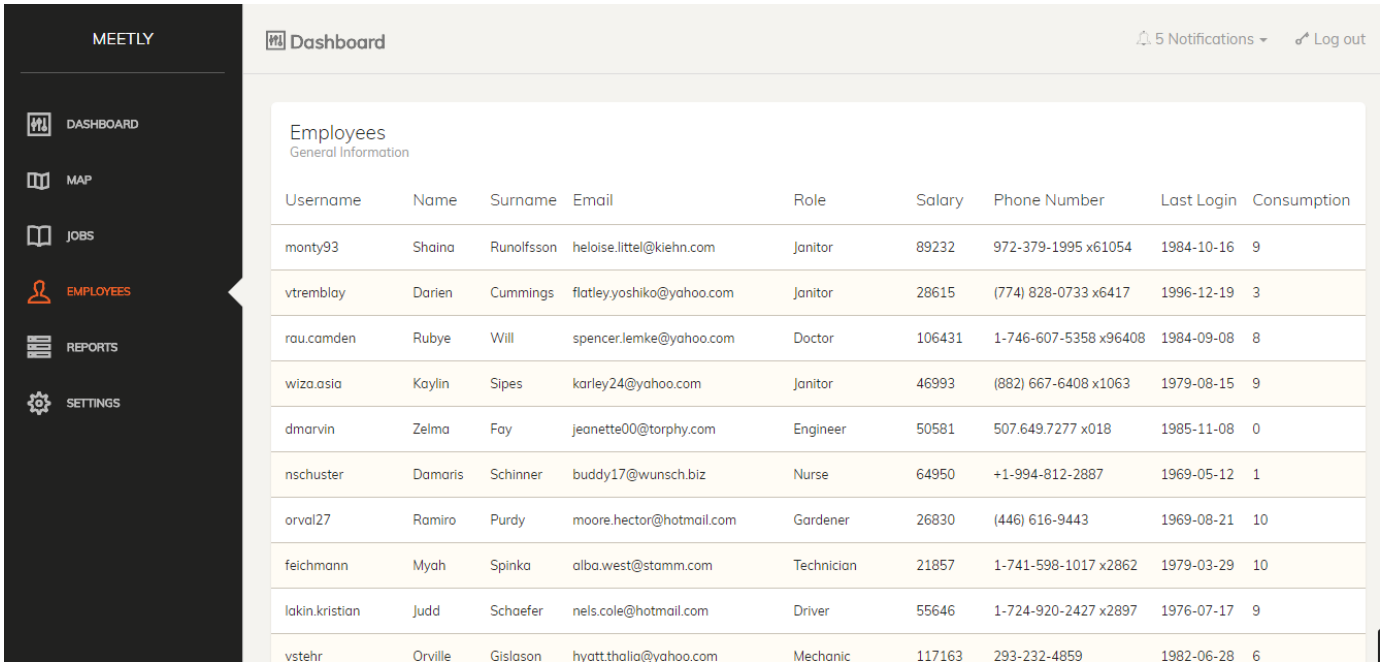
Dashboard 2



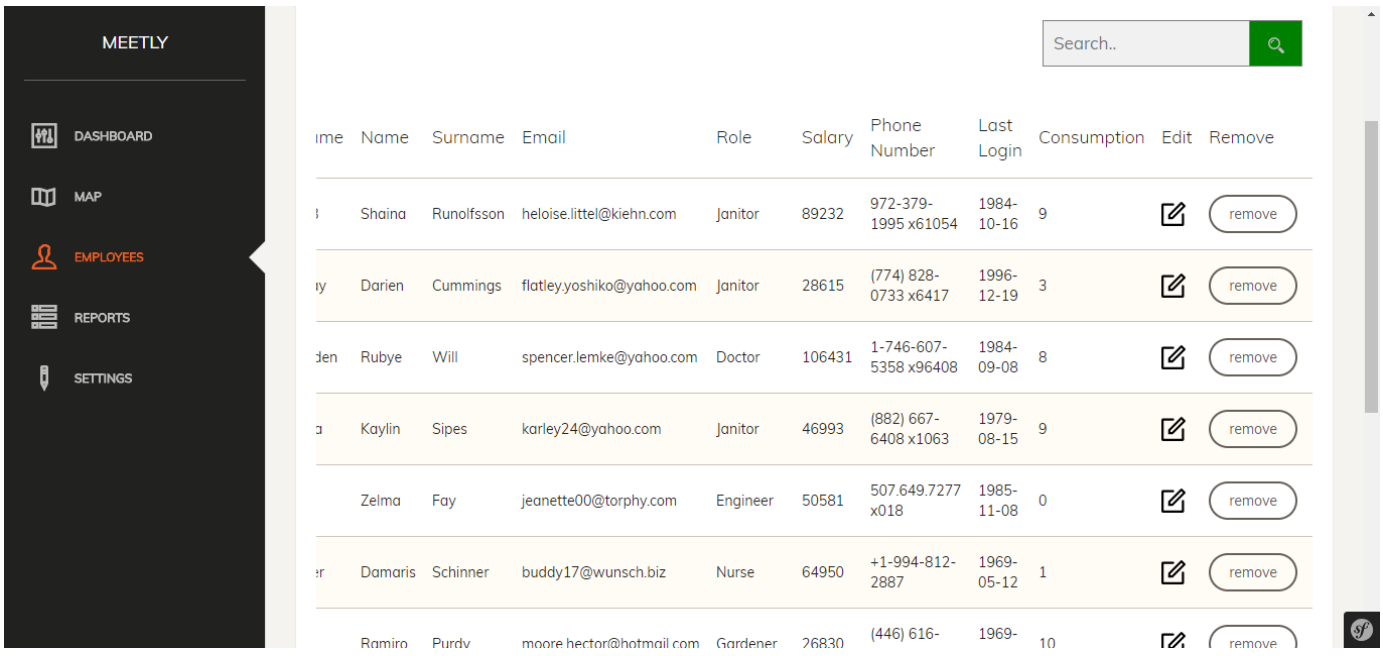
Dashboard 3



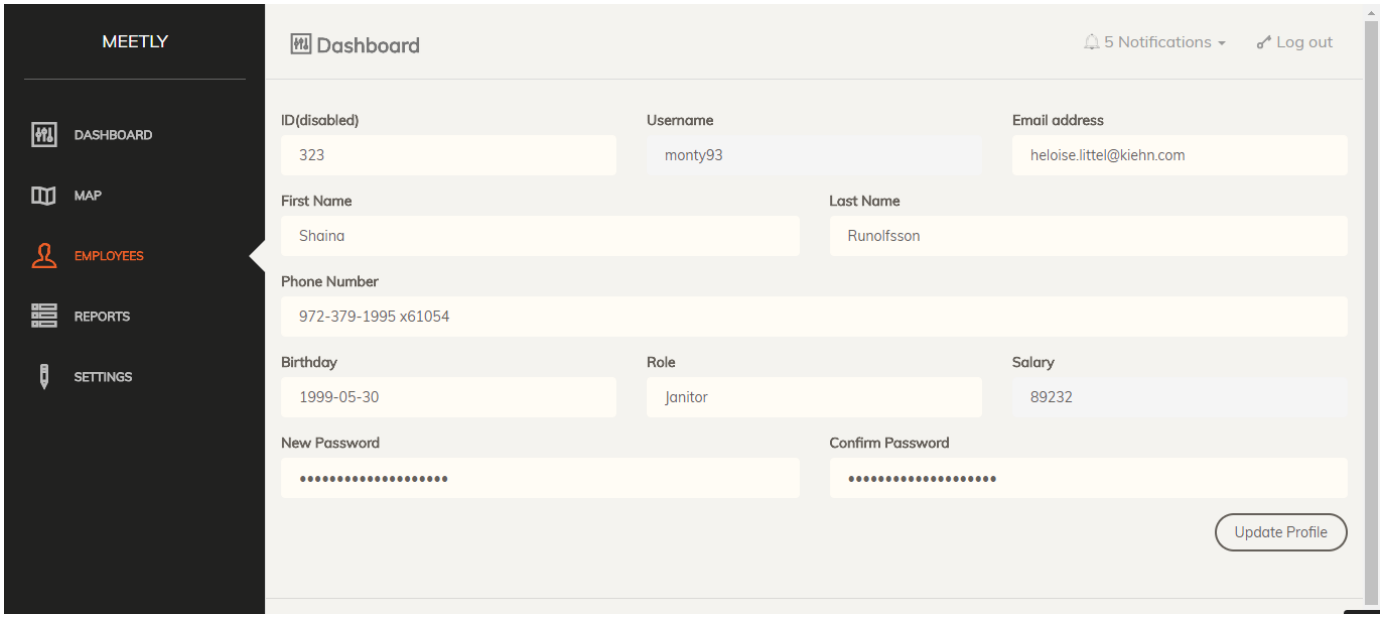
Employee 1



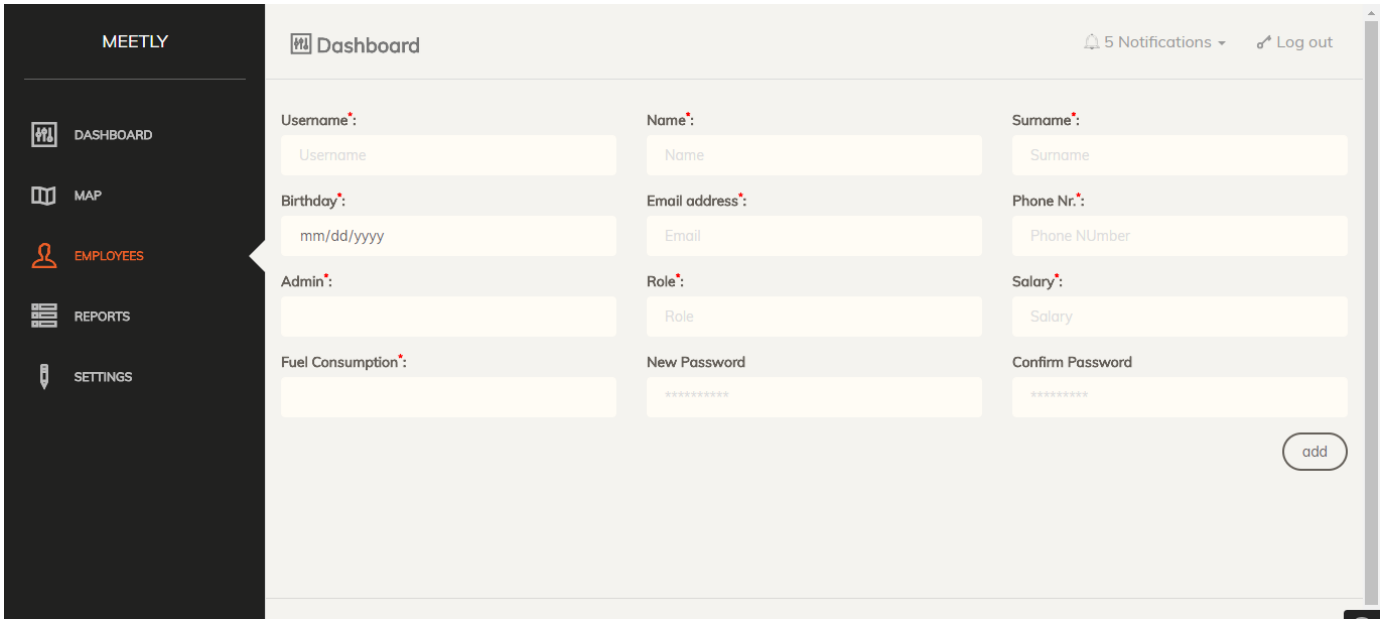
Employee 2



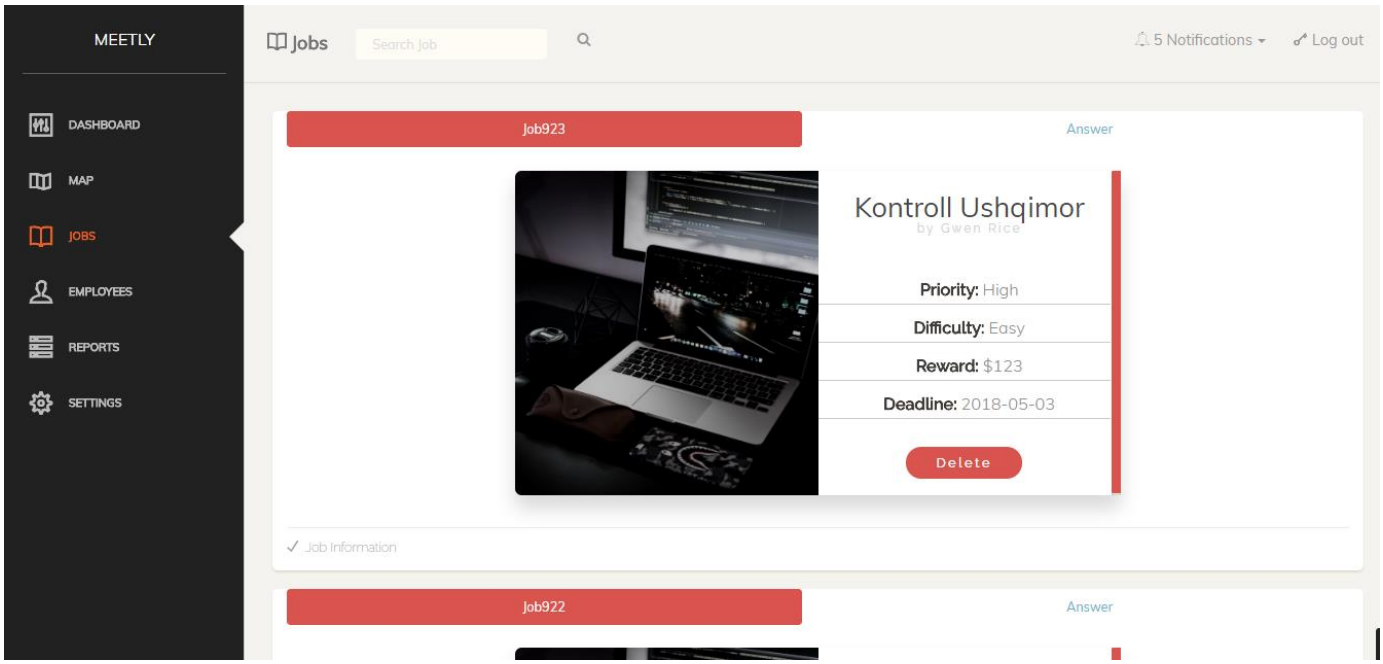
Employee 3



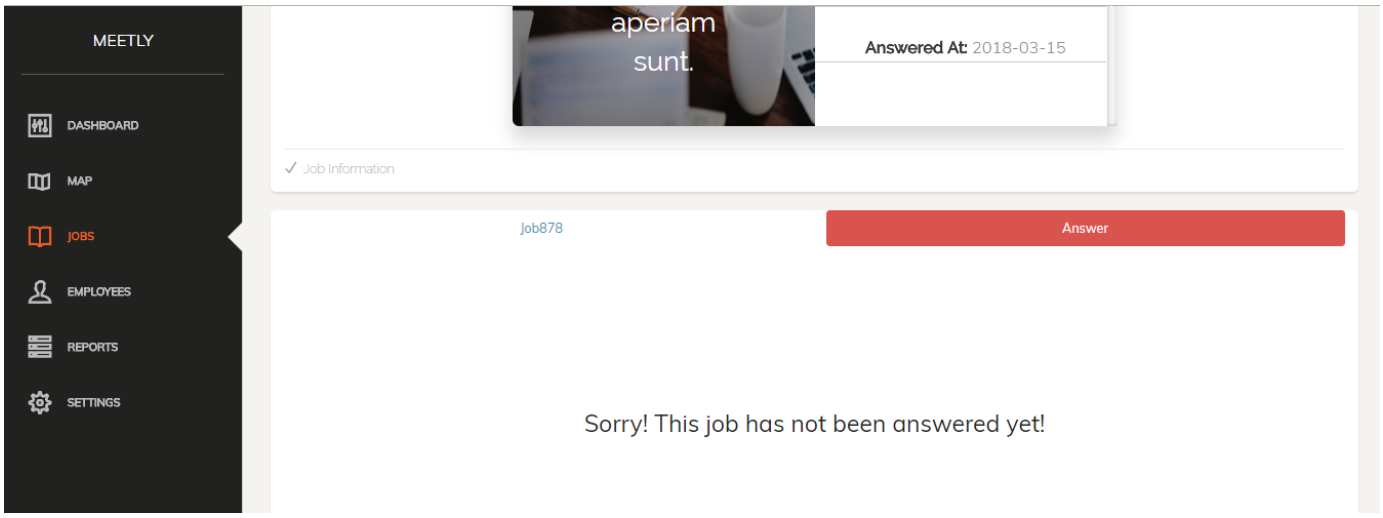
Employee 4



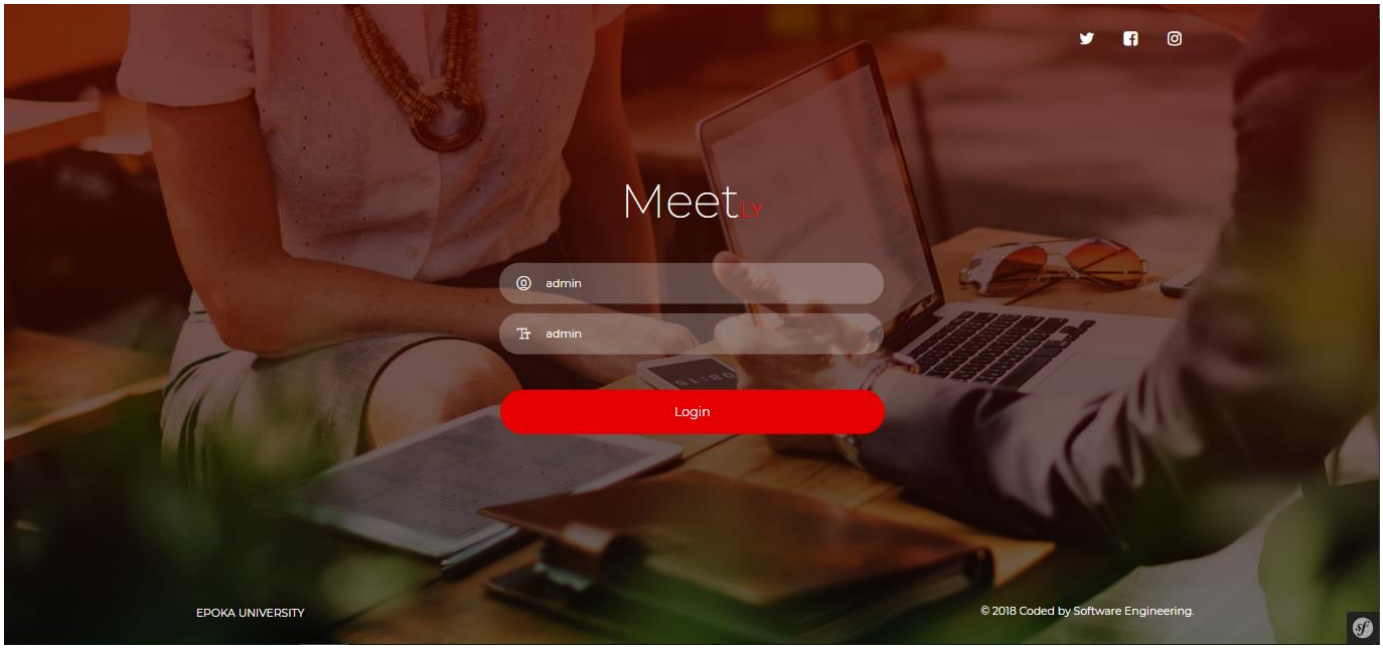
Jobs 1


MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Jobs
Search Job
5 Notifications
Log out
Job923
Answer
Kontroll Ushqimor
by Gwen Rice
Priority: High
Difficulty: Easy
Reward: $123
Deadline: 2018-05-03
Delete
Job Information
Job922
Answer

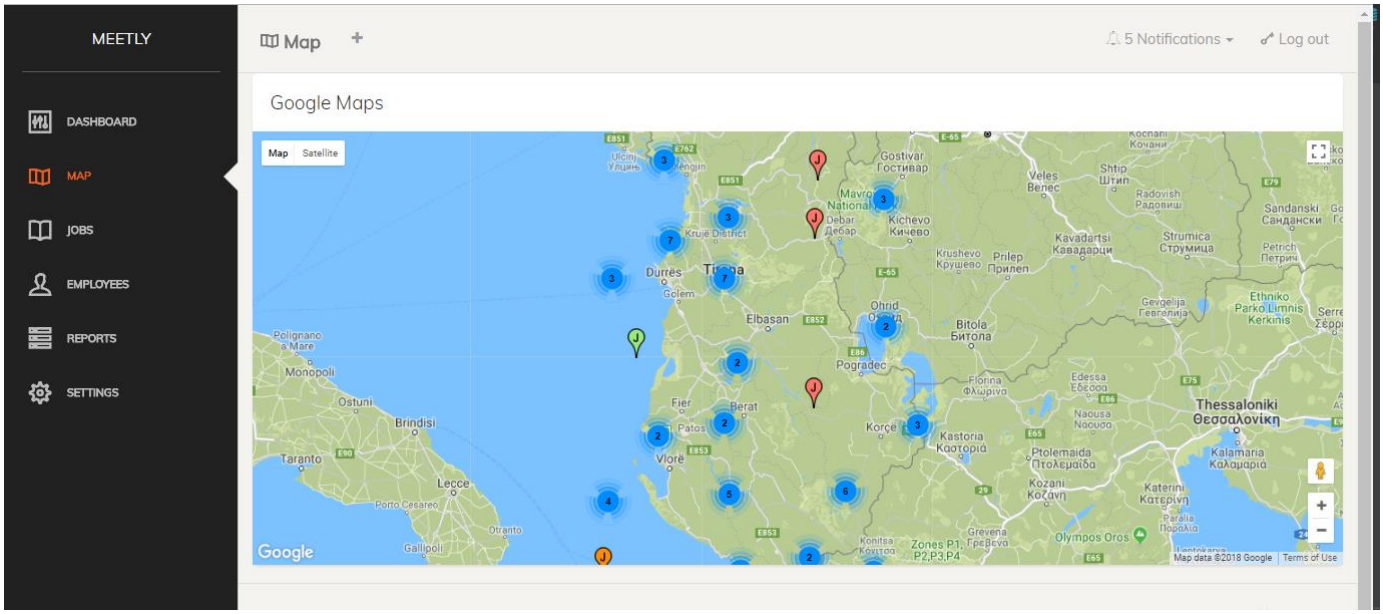
Jobs 2


MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
aperiam
sunt.
Answered At: 2018-03-15
Job Information
Job878
Answer
Sorry! This job has not been answered yet!

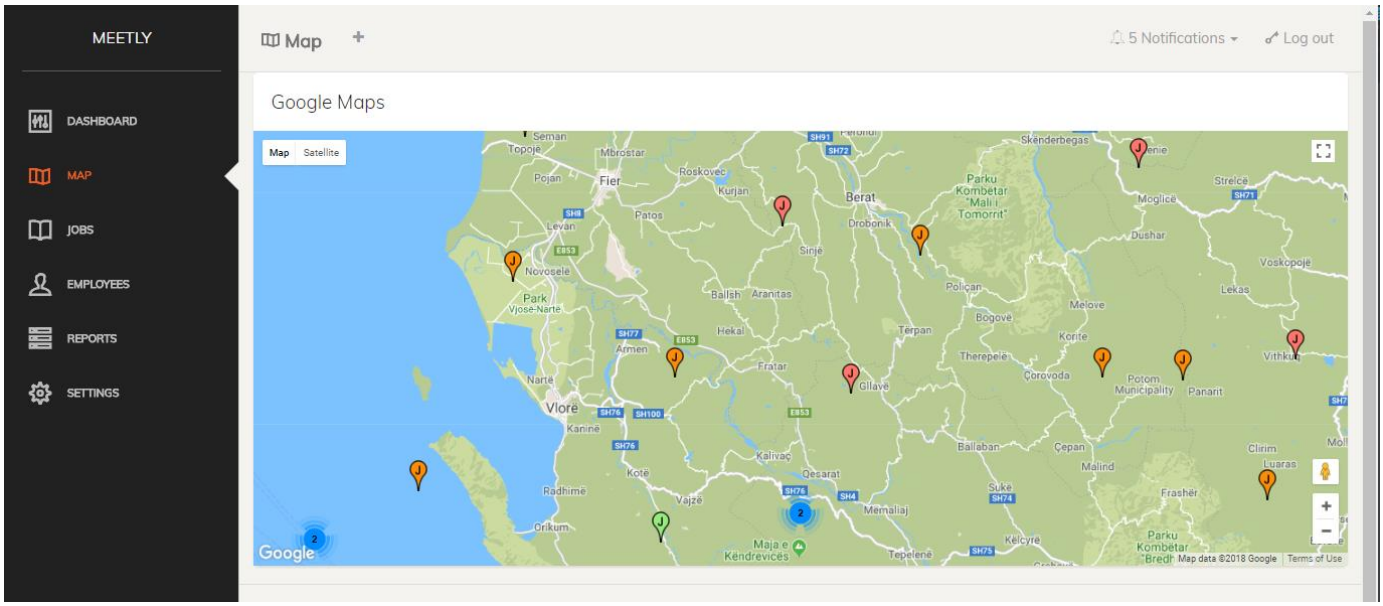
Login Page



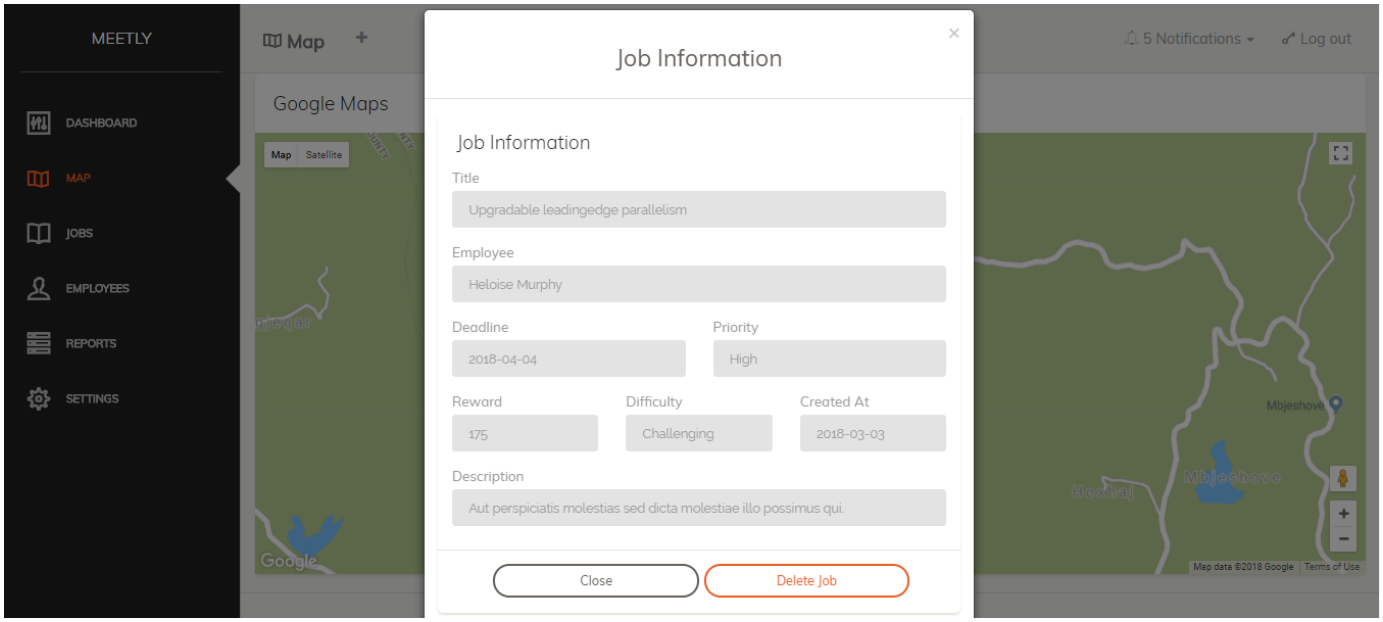
Map 1



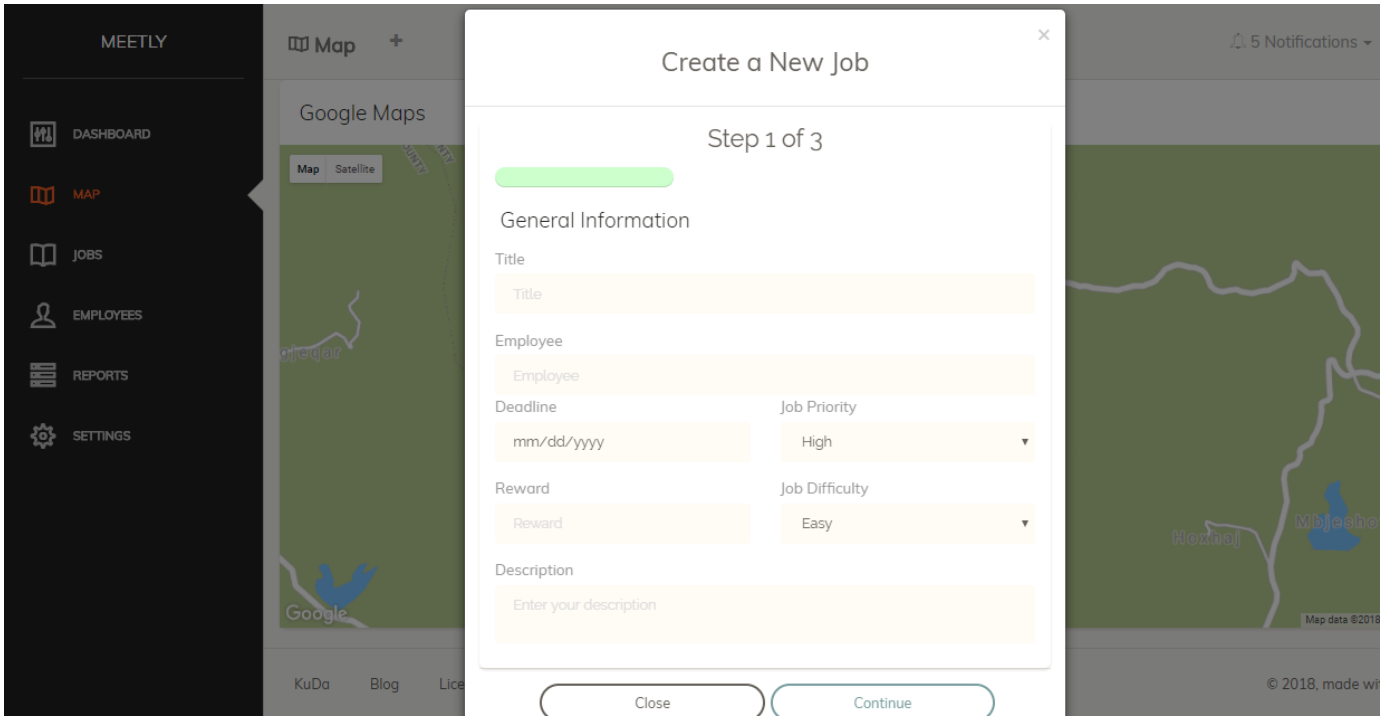
Map 2


MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Map
5 Notifications
Log out
Google Maps
Map
Satellite
Google
Map data ©2018 Google
Terms of Use

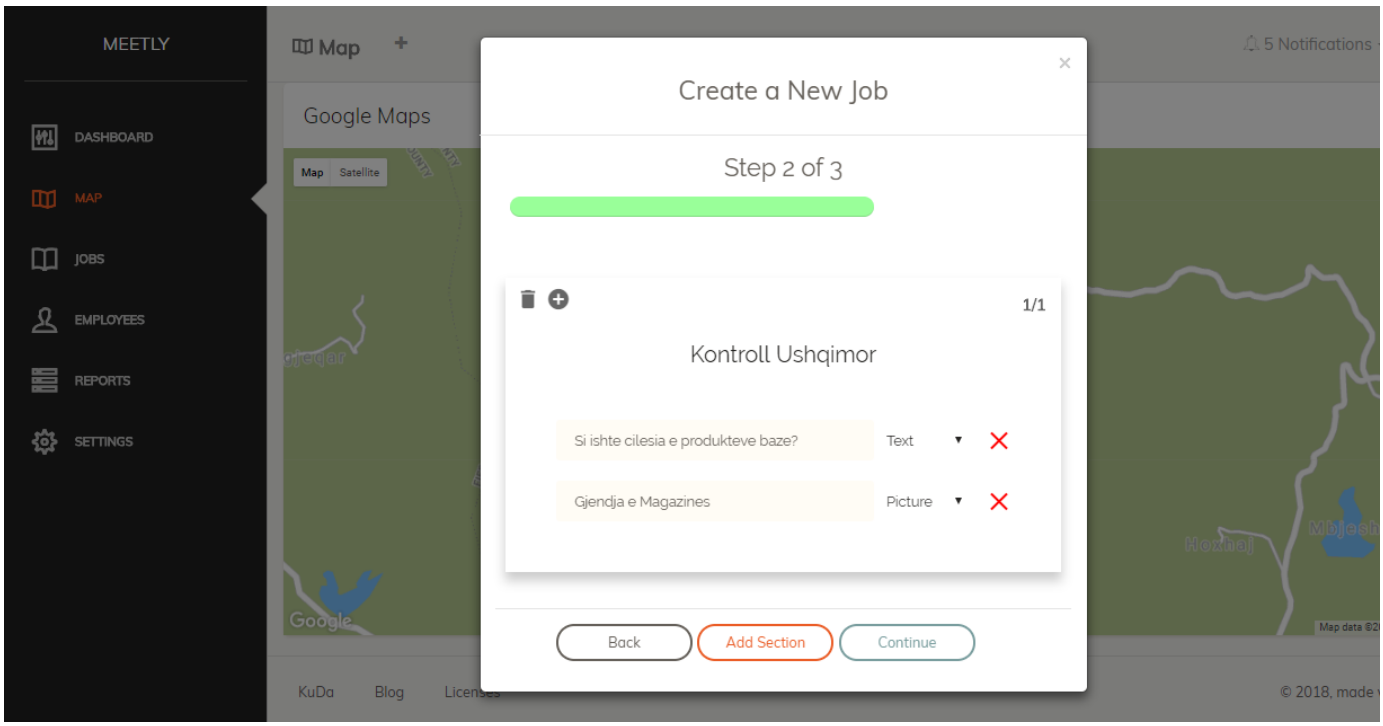
Map 3


MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Map
5 Notifications
Log out
Google Maps
Job Information
Job Information
Title
Upgradable leadingedge parallelism
Employee
Heloise Murphy
Deadline
2018-04-04
Priority
High
Reward
175
Difficulty
Challenging
Created At
2018-03-03
Description
Aut perspiciatis molestias sed dicta molestiae illo possimus qui.
Close
Delete Job

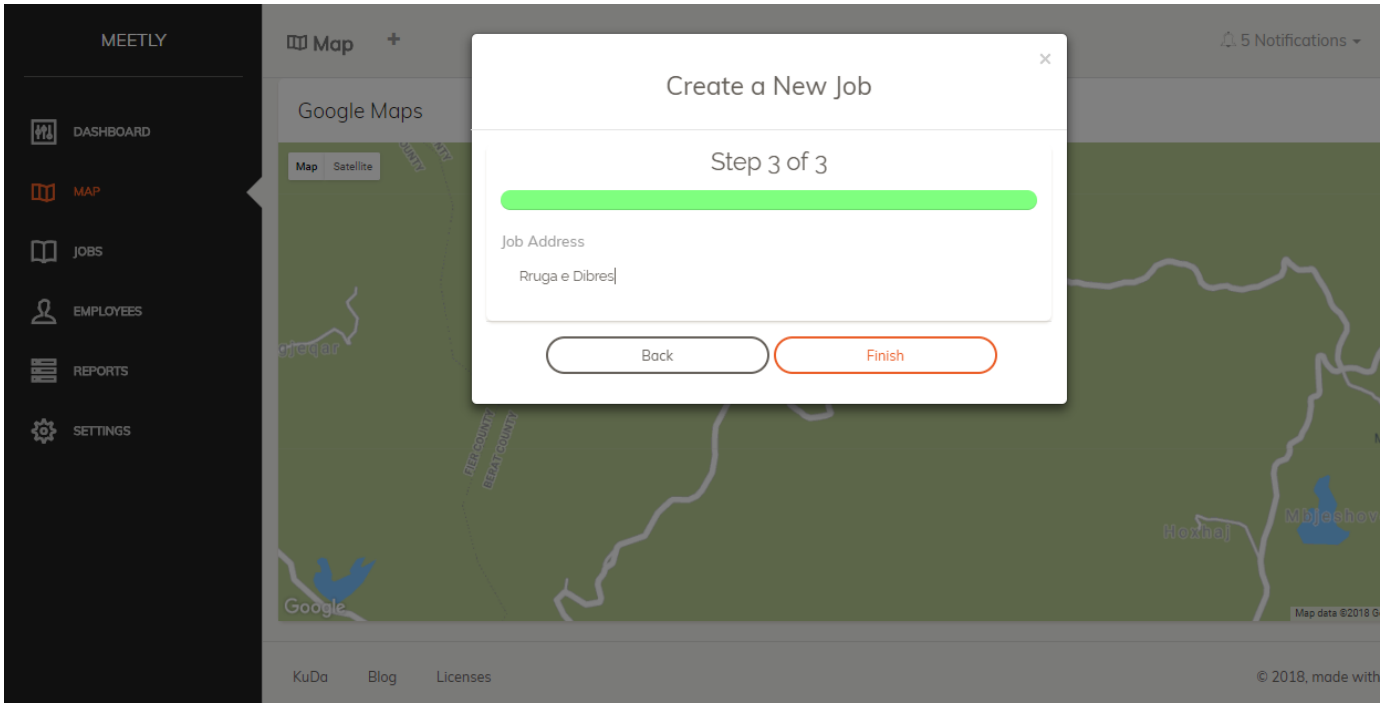
Map 4


MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Map
5 Notifications
Google Maps
Map
Satellite
Create a New Job
Step 1 of 3
General Information
Title
Employee
Deadline
mm/dd/yyyy
Job Priority
High
Reward
Job Difficulty
Easy
Description
Enter your description
Close
Continue
KuDa
Blog

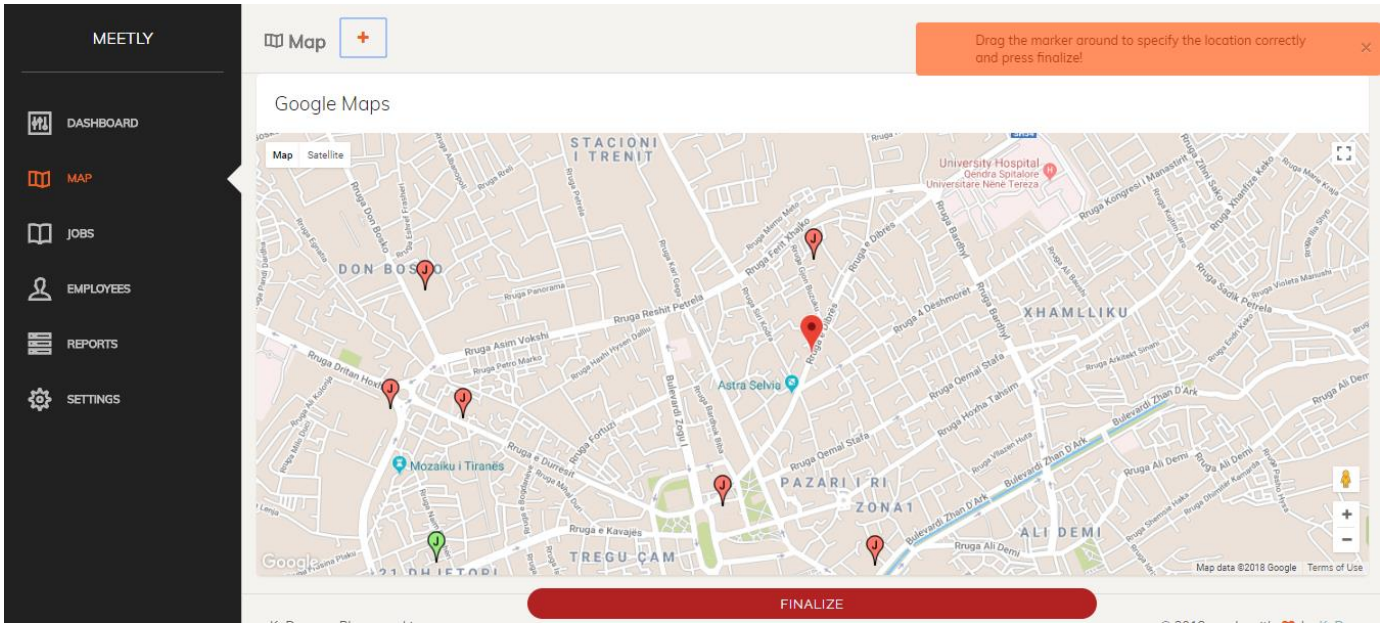
Map 5


MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Map
5 Notifications
Google Maps
Map
Satellite
Create a New Job
Step 2 of 3
1/1
Kontroll Ushqimor
Si ishte cilesia e produkteve baze?
Text
Gjendja e Magazines
Picture
Back
Add Section
Continue
KuDa
Blog

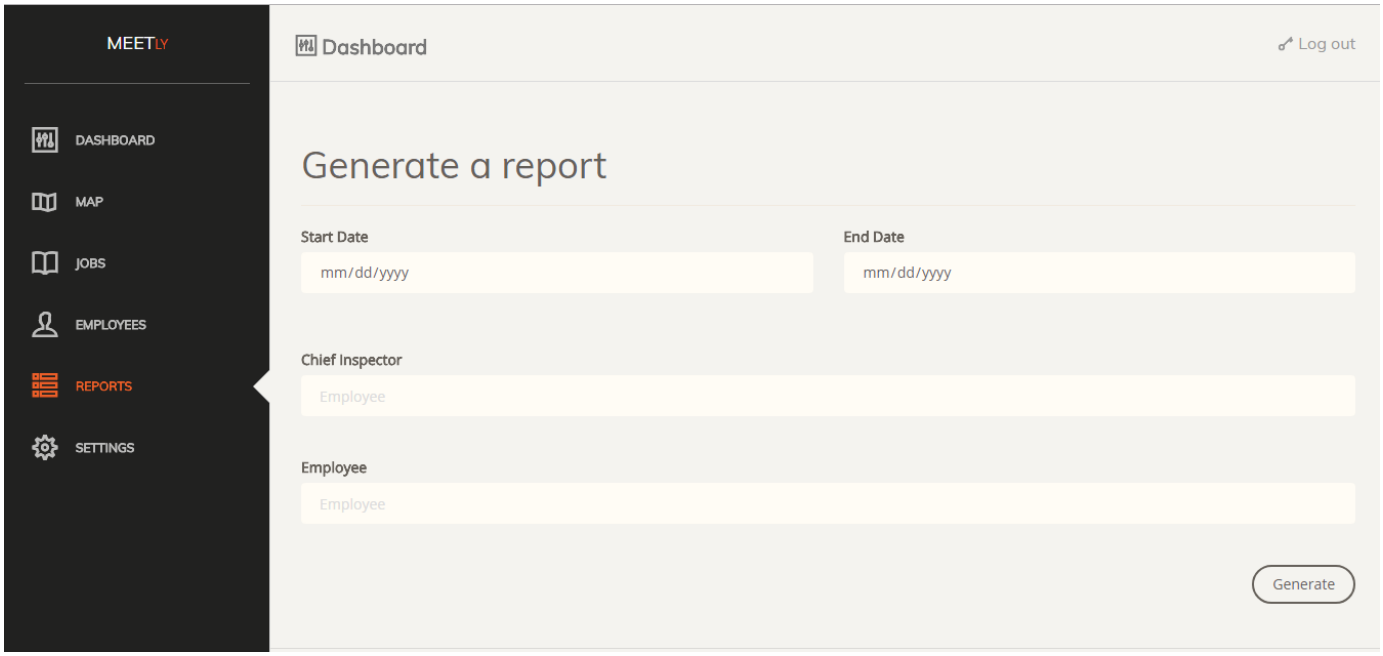
Map 6


MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Map
5 Notifications
Google Maps
Map
Satellite
Create a New Job
Step 3 of 3
Job Address
Rruga e Dibres
Back
Finish
KuDa
Blog
Licenses
© 2018, made with

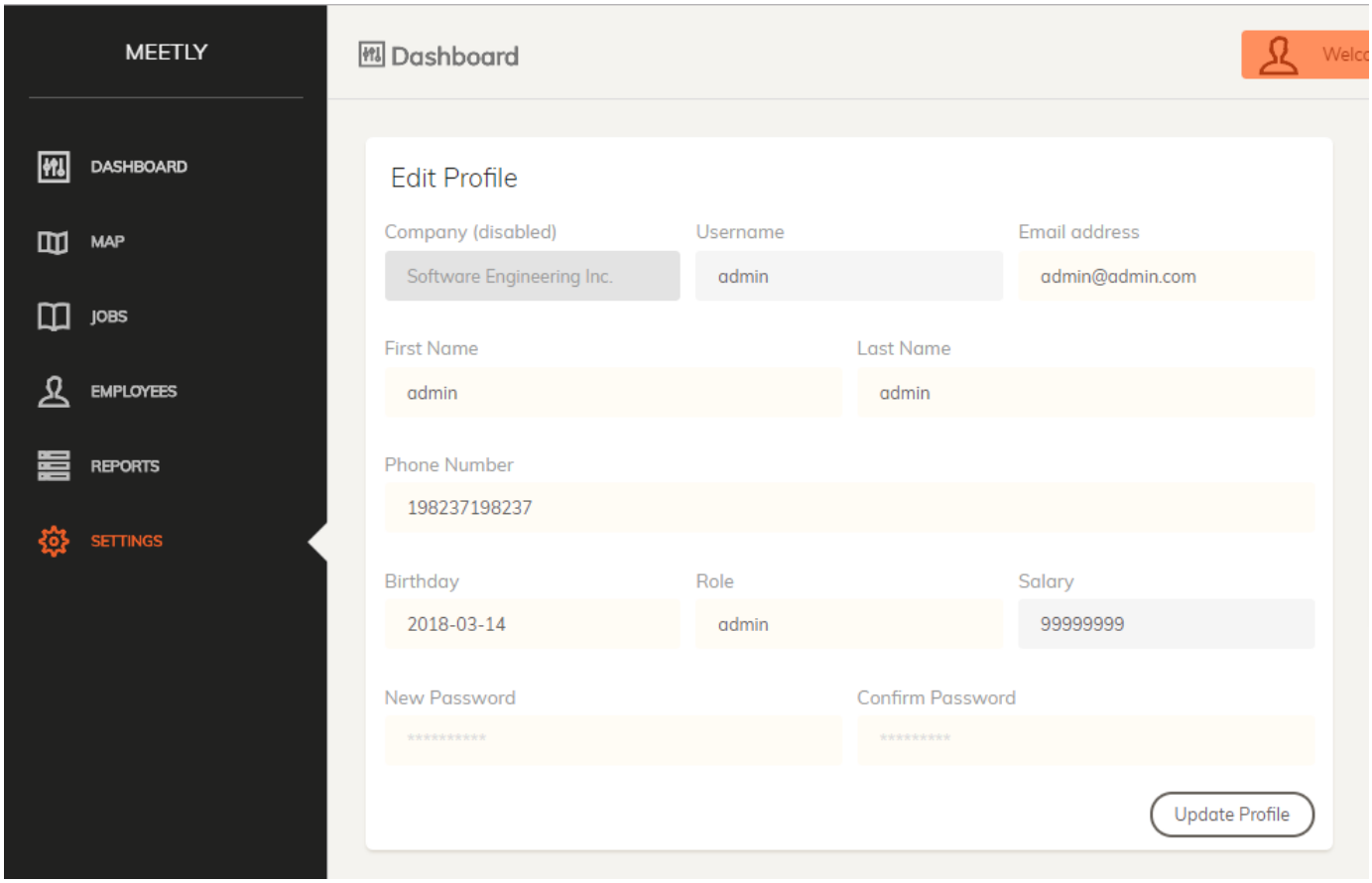
Map 7


MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Map
Drag the marker around to specify the location correctly and press finalize!
Google Maps
Map
Satellite
STACIONI I TRENIT
DON BOSKO
XHAMLLIKU
PAZARI I RI
ZONA 1
ALI DEMI
TREGU CAM
Astra Selvia
Mozaiku i Tiranës
University Hospital
FINALIZE

Reports 1

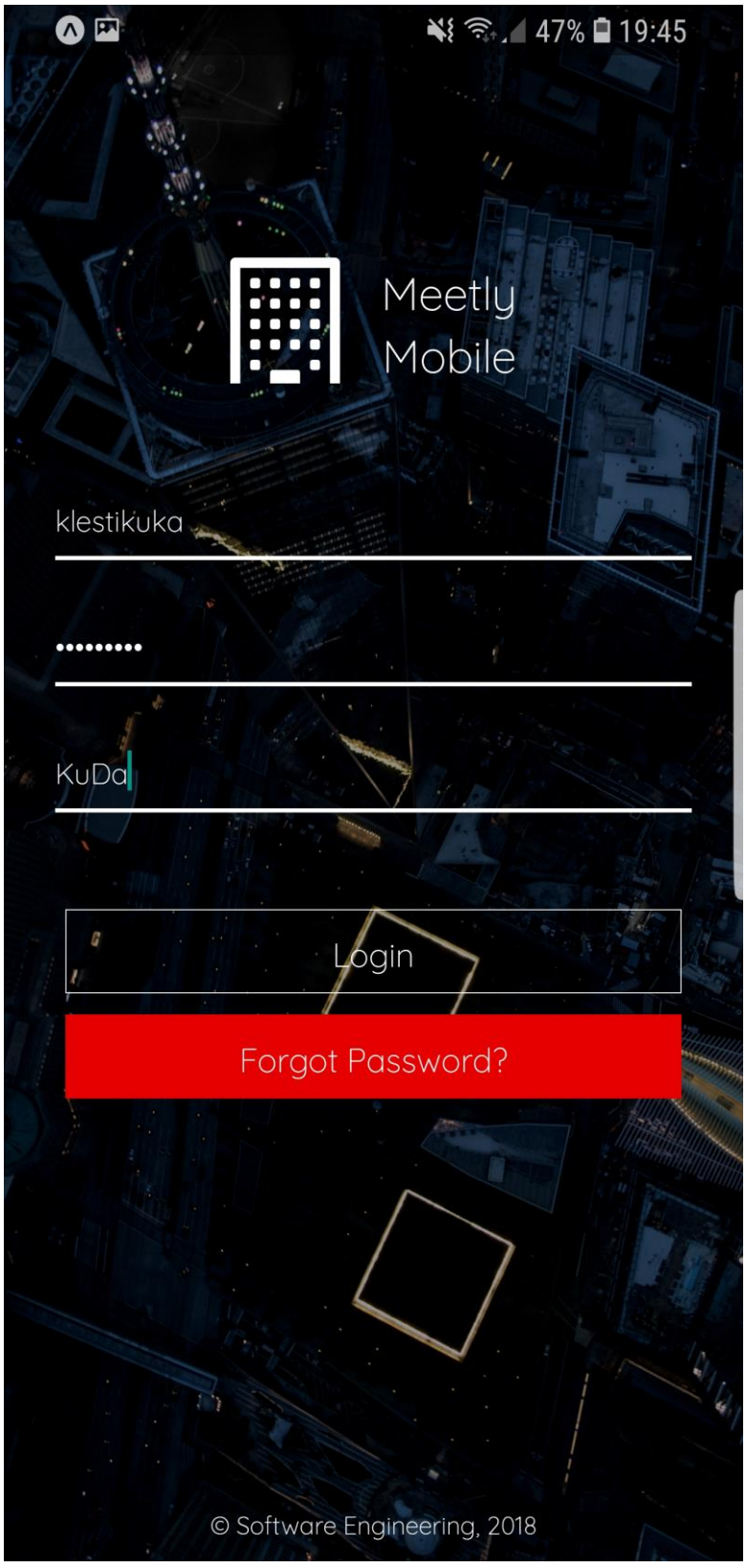

MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Dashboard
Log out
Generate a report
Start Date
mm/dd/yyyy
End Date
mm/dd/yyyy
Chief Inspector
Employee
Employee
Employee
Generate

Settings

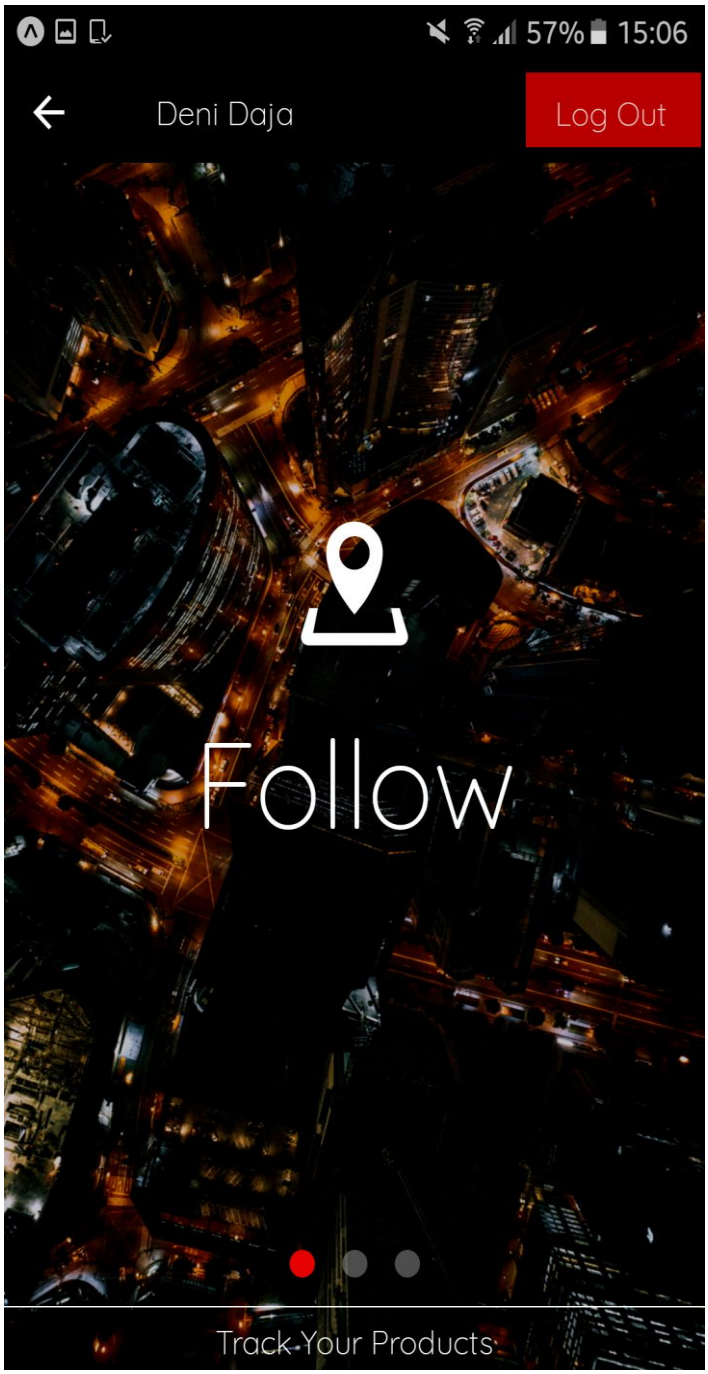

MEETLY
DASHBOARD
MAP
JOBS
EMPLOYEES
REPORTS
SETTINGS
Dashboard
Edit Profile
Company (disabled)
Software Engineering Inc.
Username
admin
Email address
admin@admin.com
First Name
admin
Last Name
admin
Phone Number
198237198237
Birthday
2018-03-14
Role
admin
Salary
99999999
New Password
Confirm Password
Update Profile

Meetly Mobile Splash Screen

Meetly Mobile Login Page

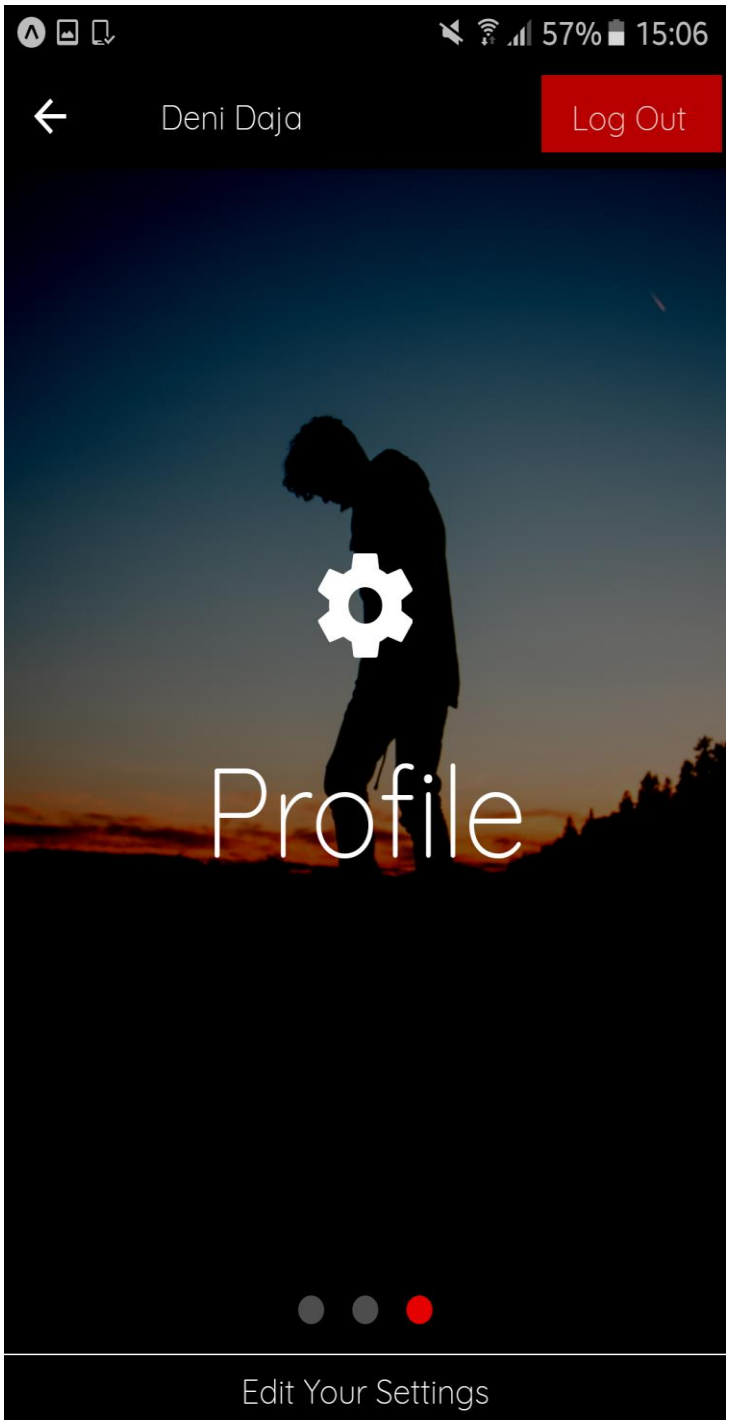


Meetly Mobile Menu 1

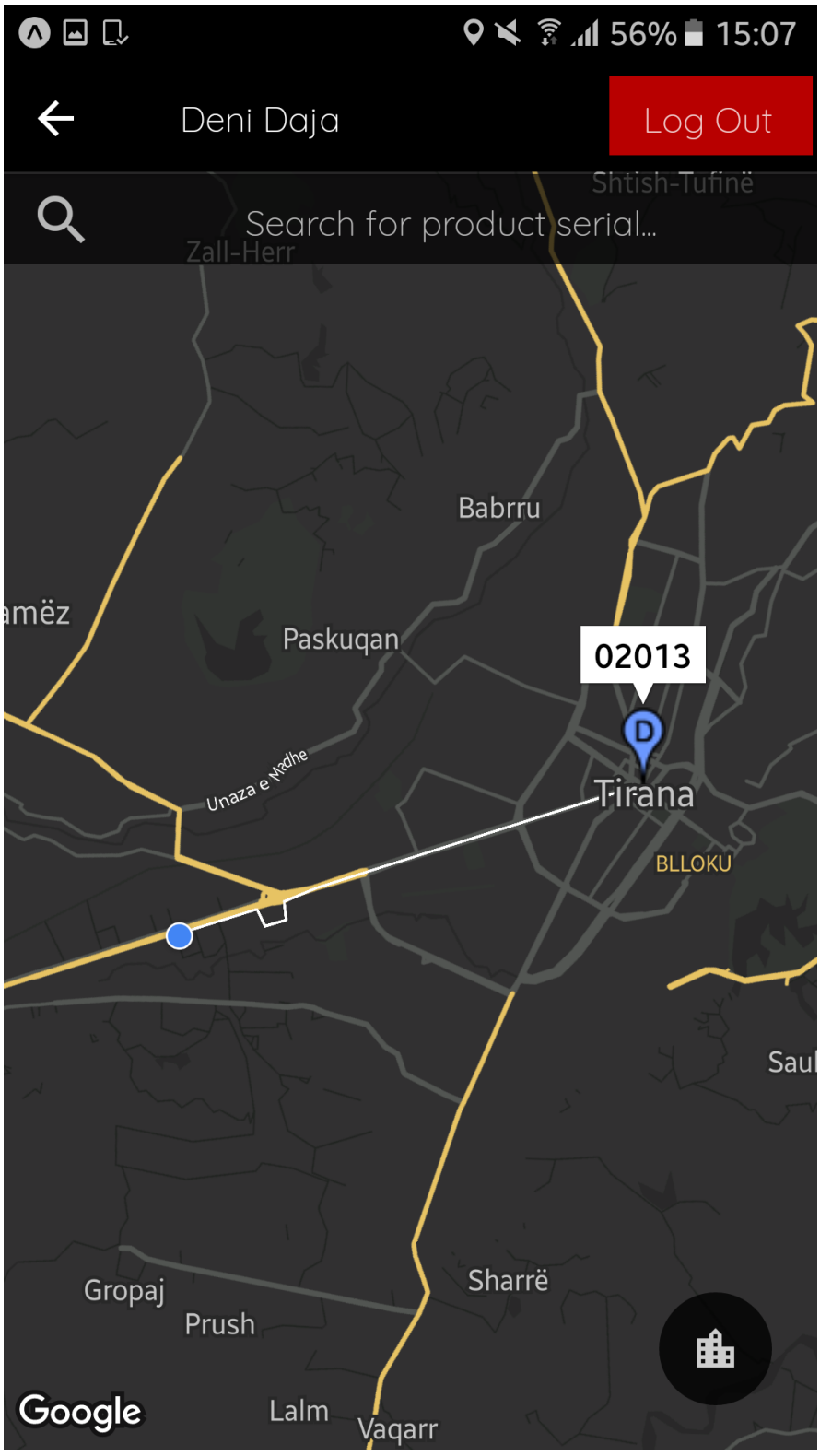


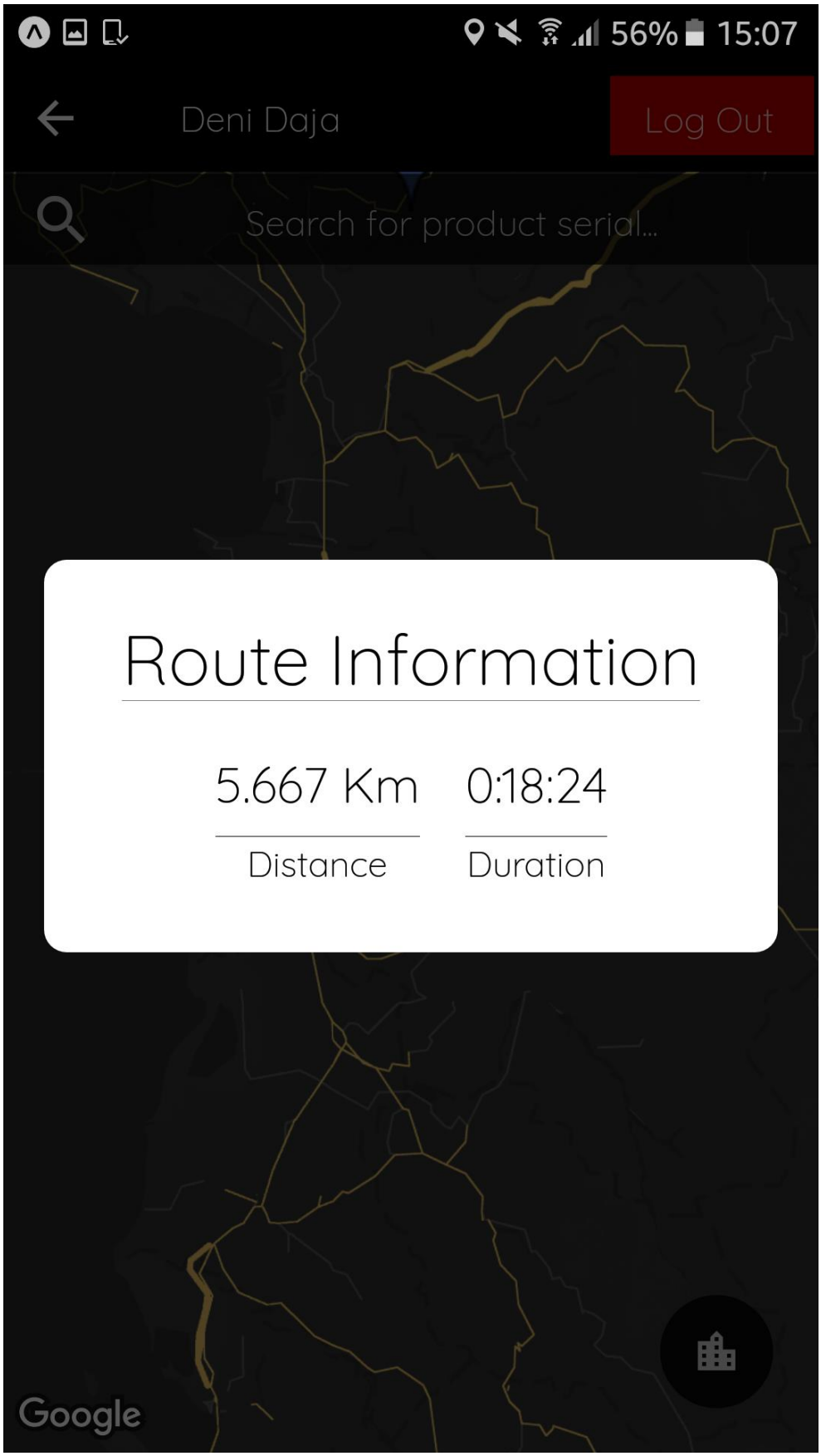
Meetly Mobile Menu 2

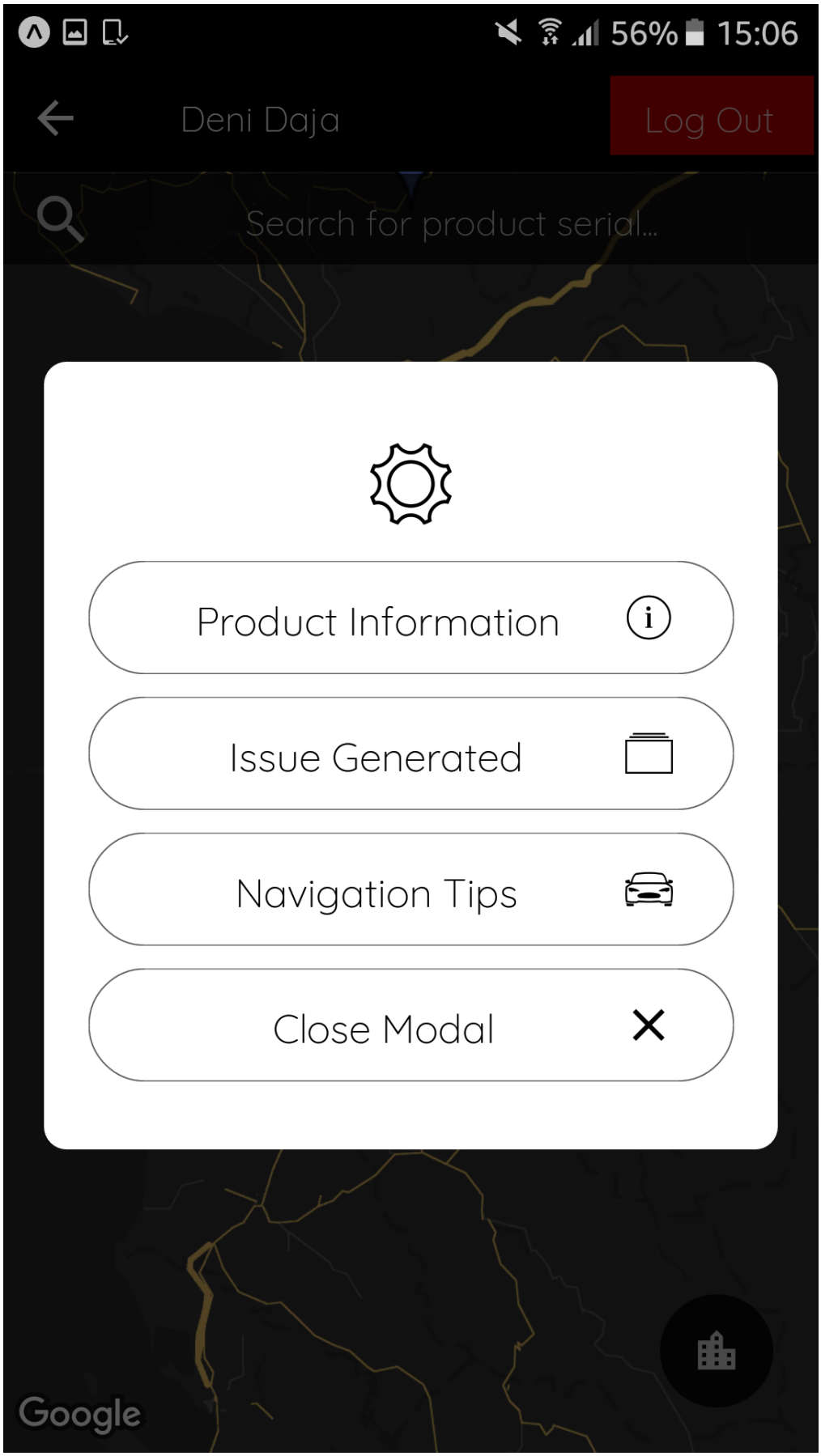
Meetly Mobile Menu 3



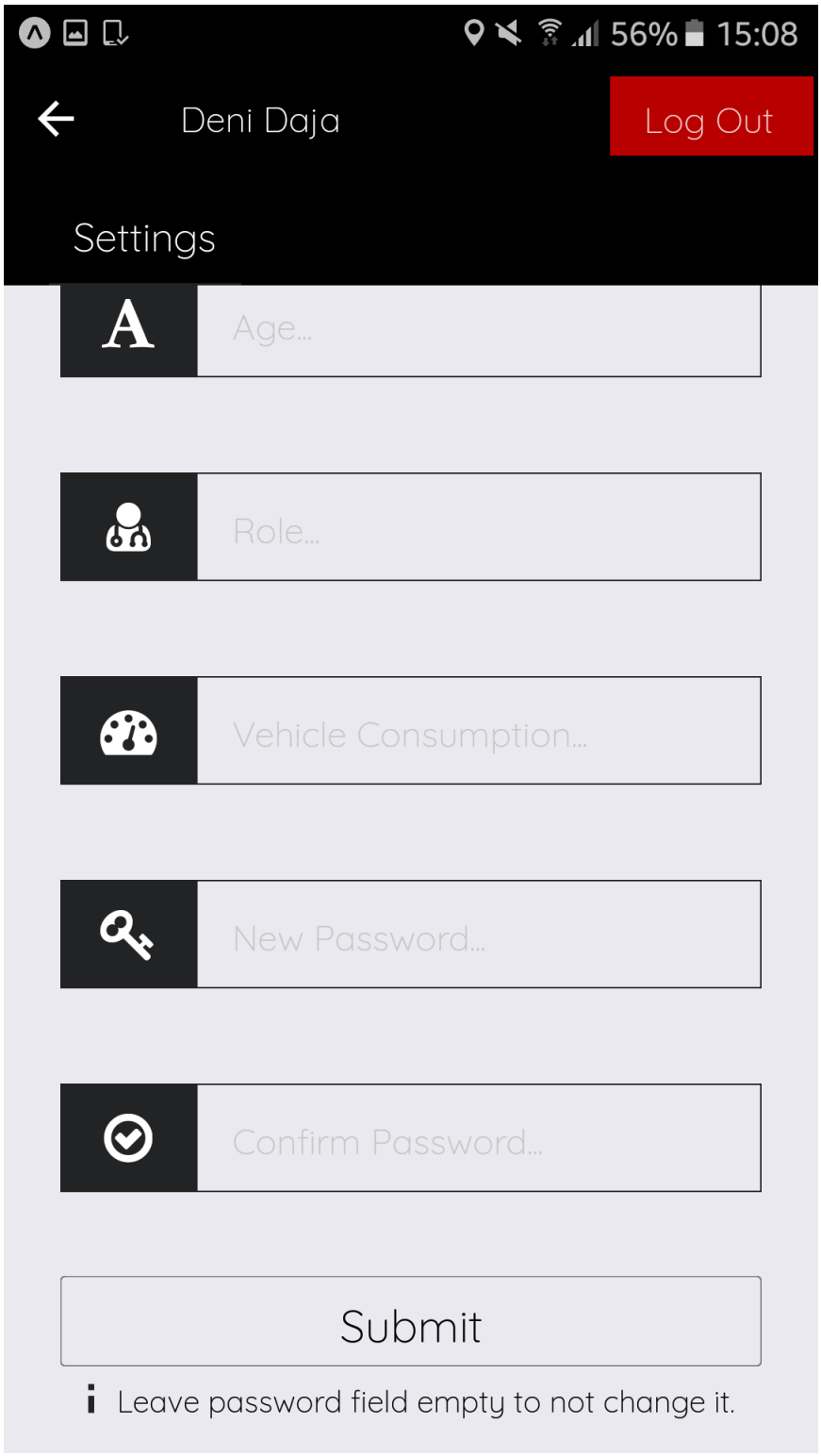
Meetly Mobile Map



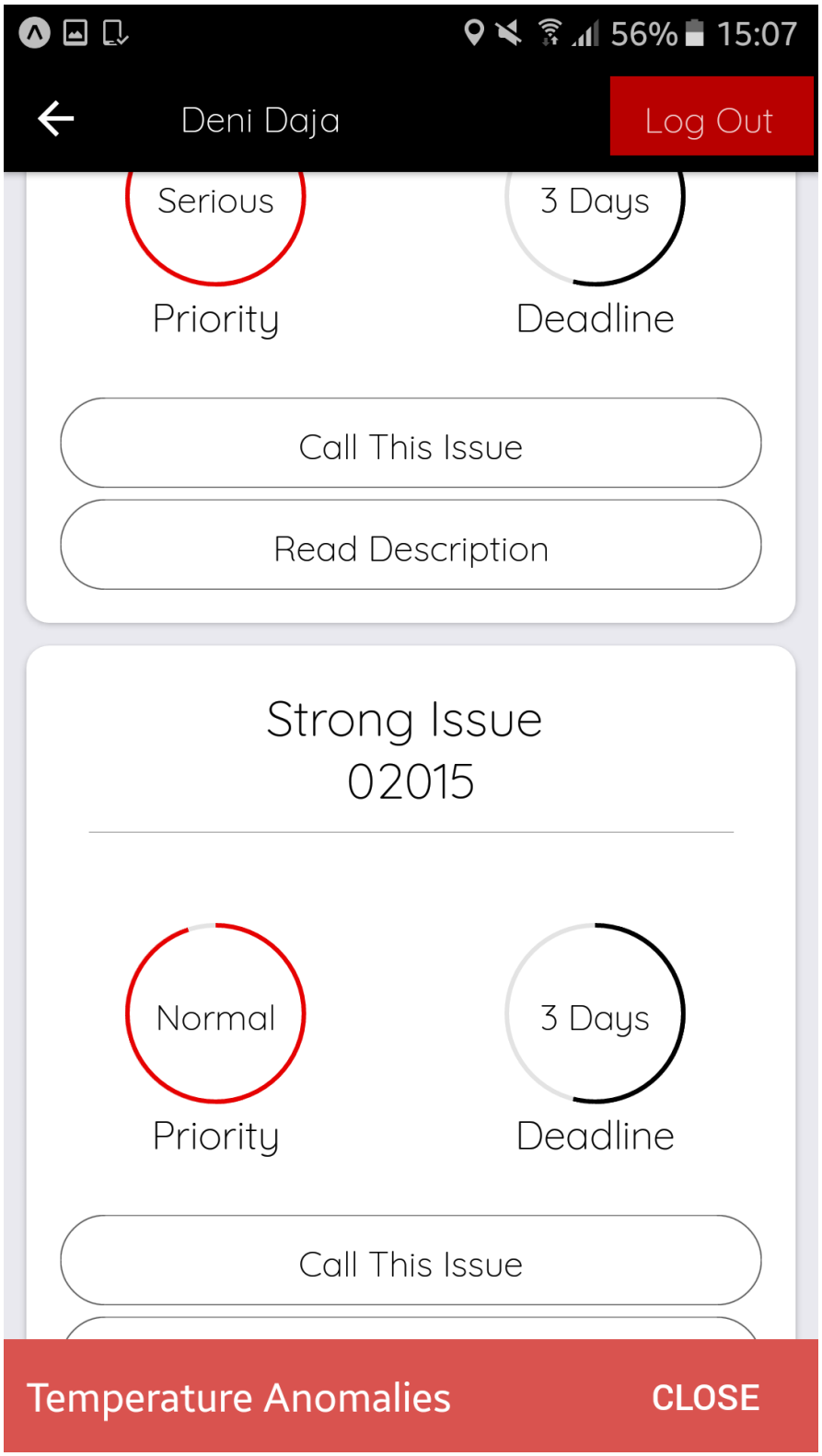

56% 15:07
Deni Daja
Log Out
Search for product serial...
Route Information
5.667 Km
0:18:24
Distance
Duration
Google

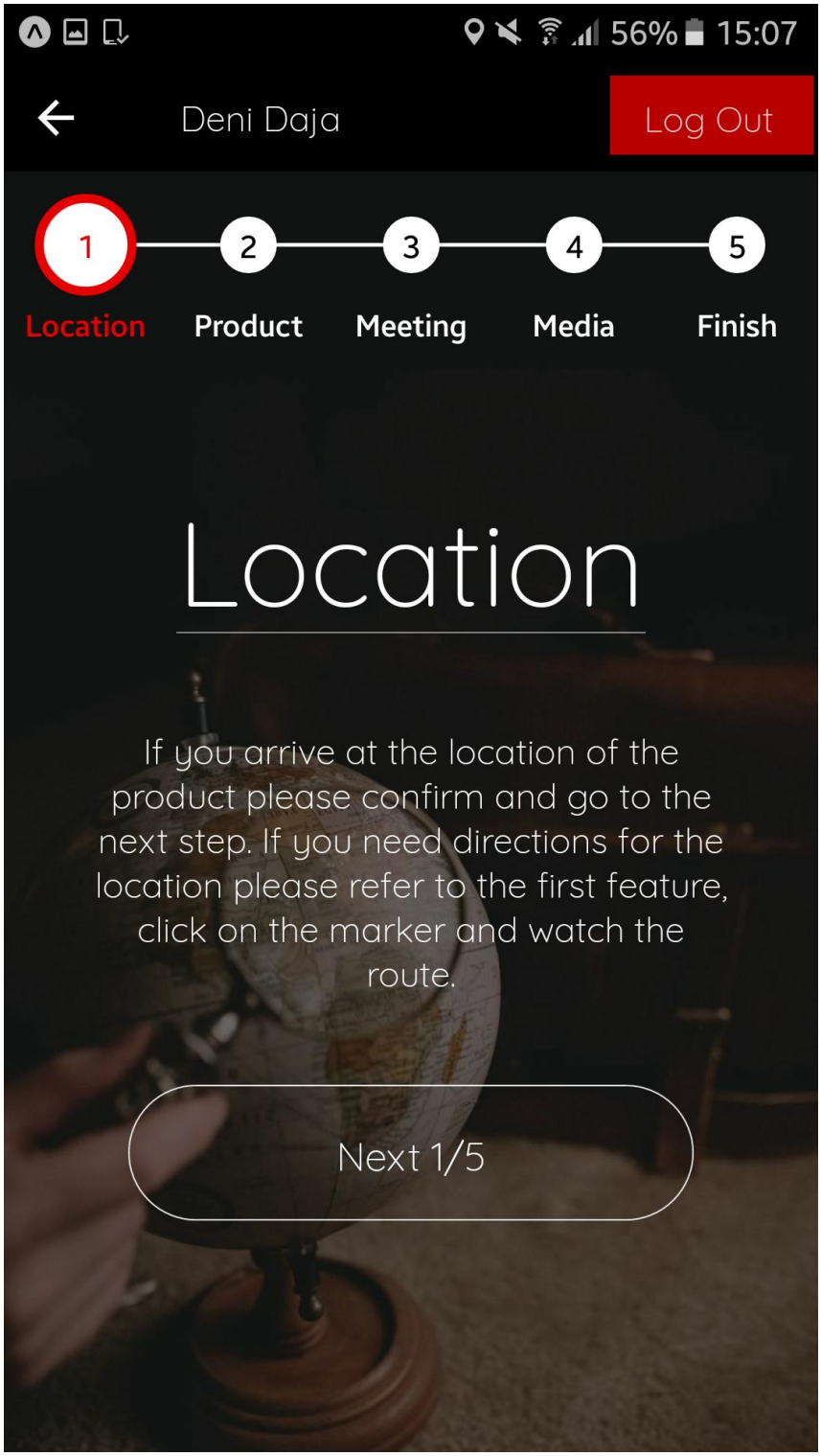

56% 15:06
Deni Daja
Log Out
Search for product serial...
Product Information
Issue Generated
Navigation Tips
Close Modal
Google

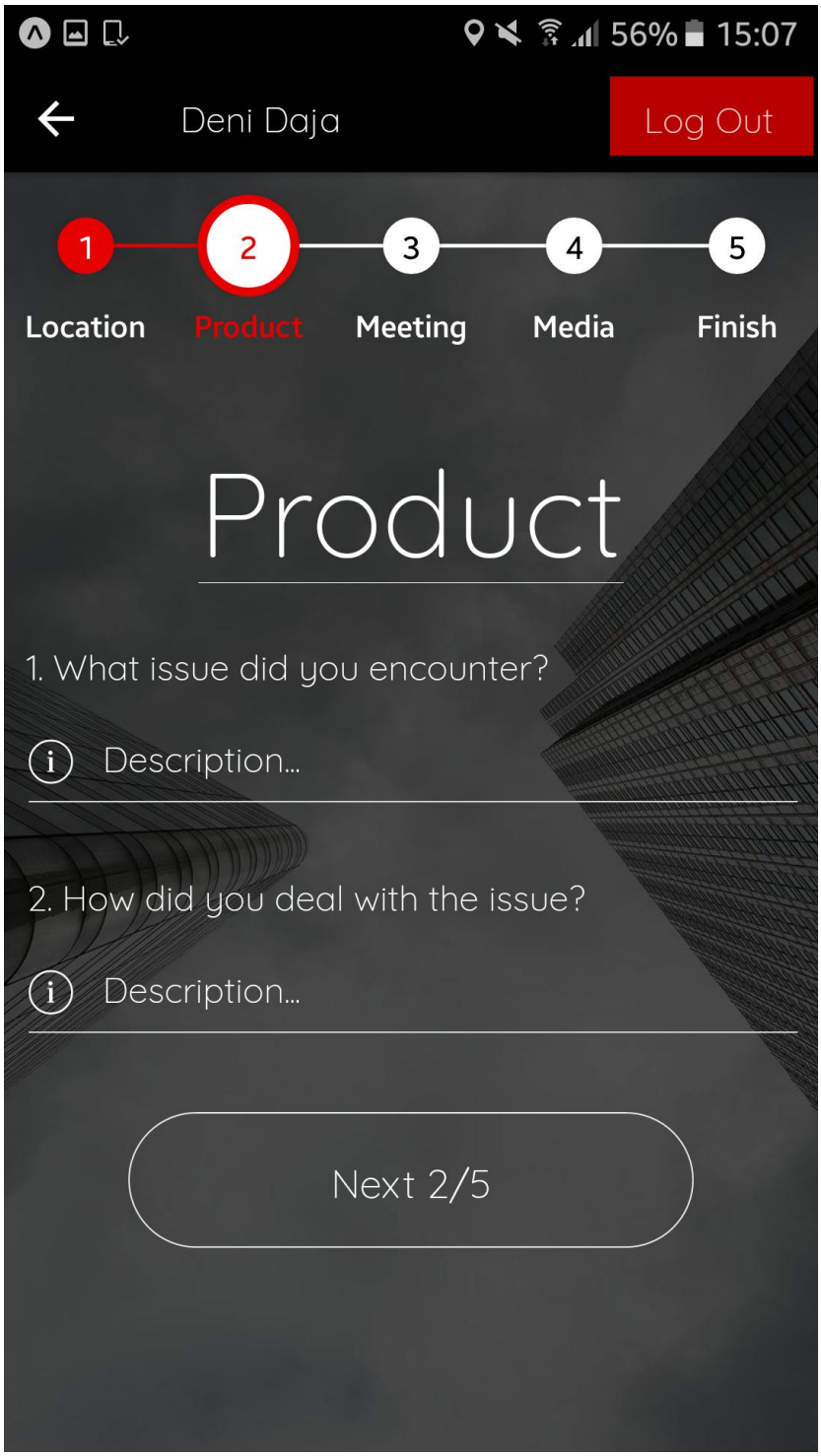
Meetly Mobile Settings Page

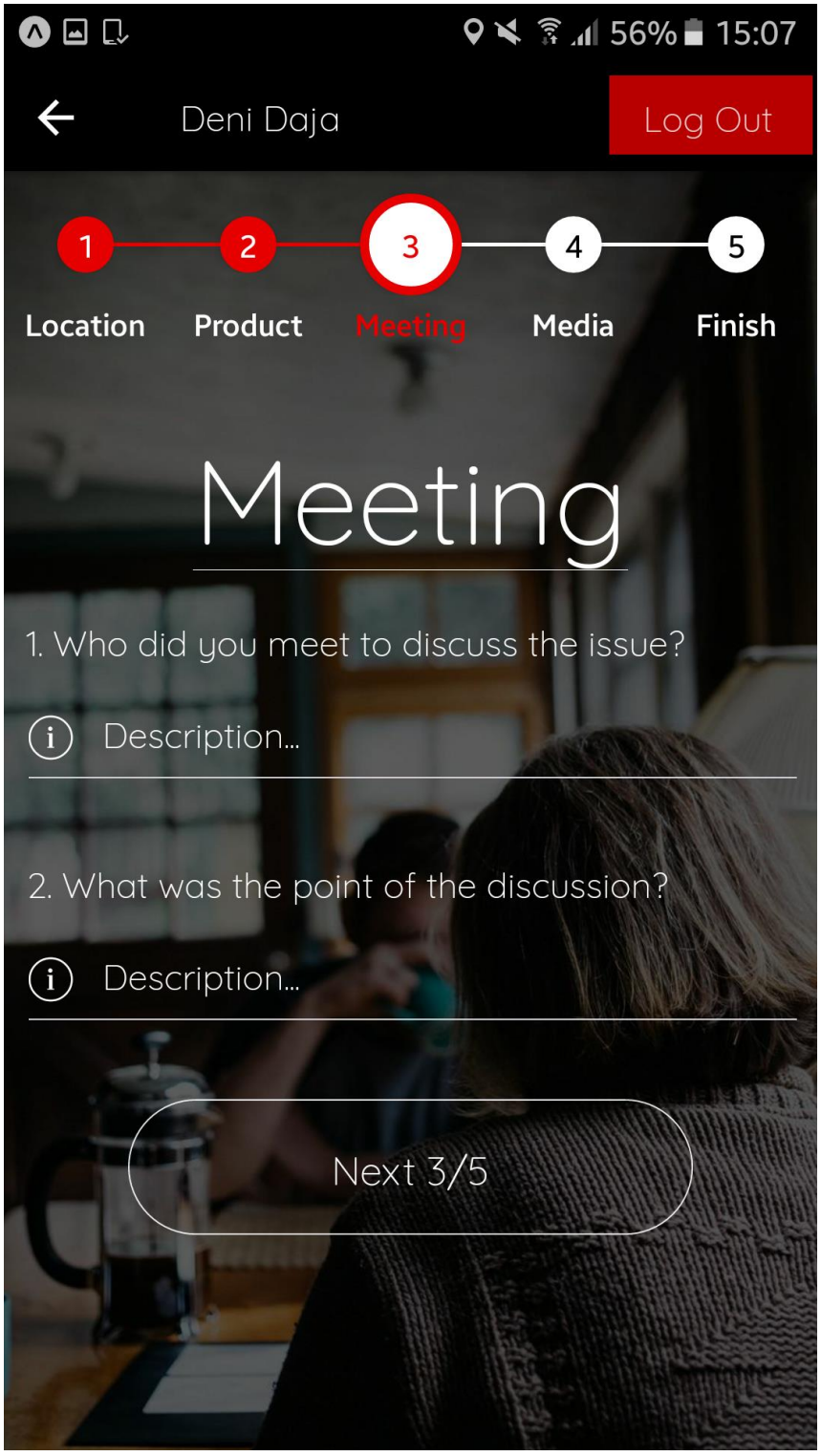


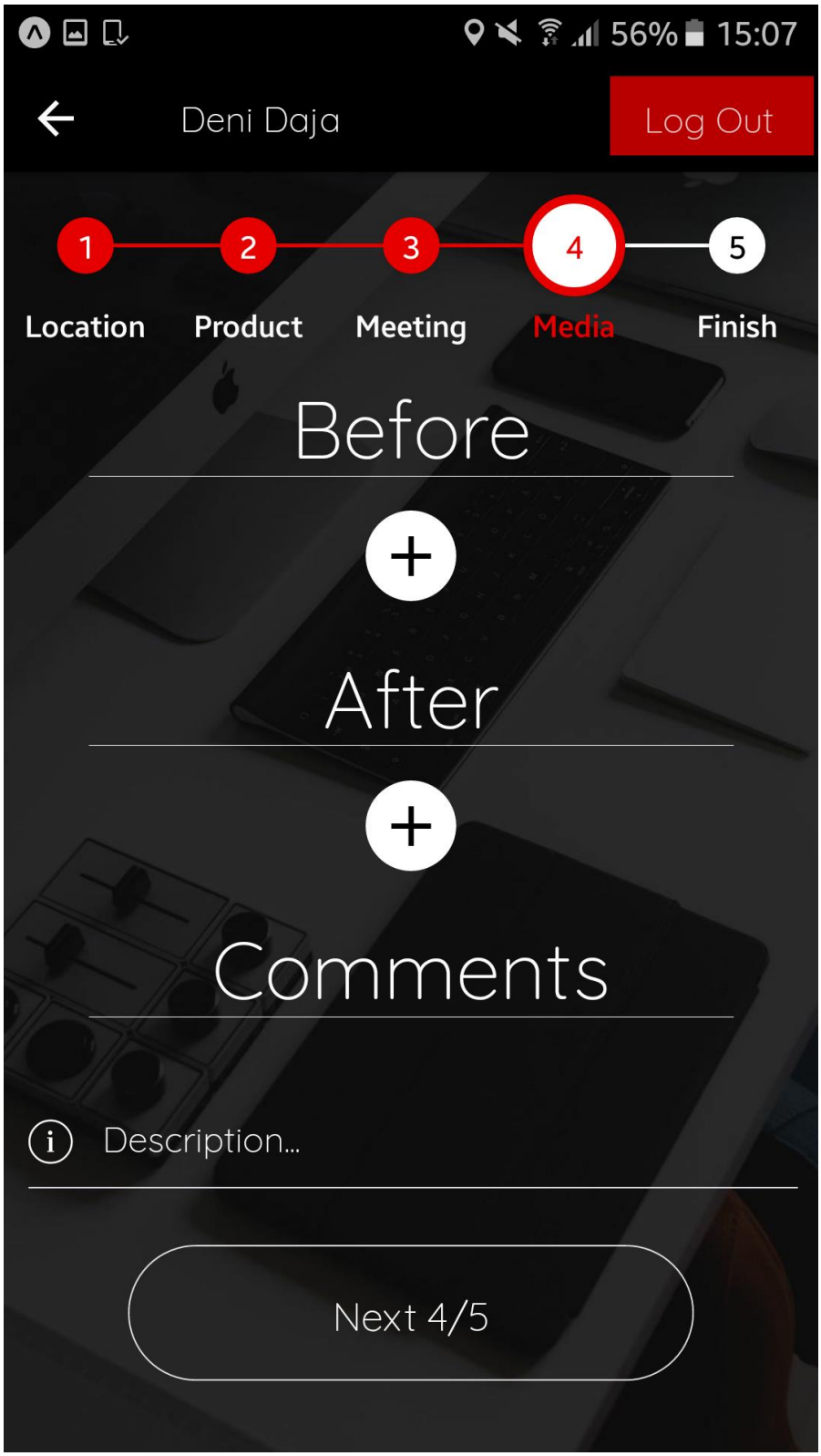
Meetly Mobile Reports Page

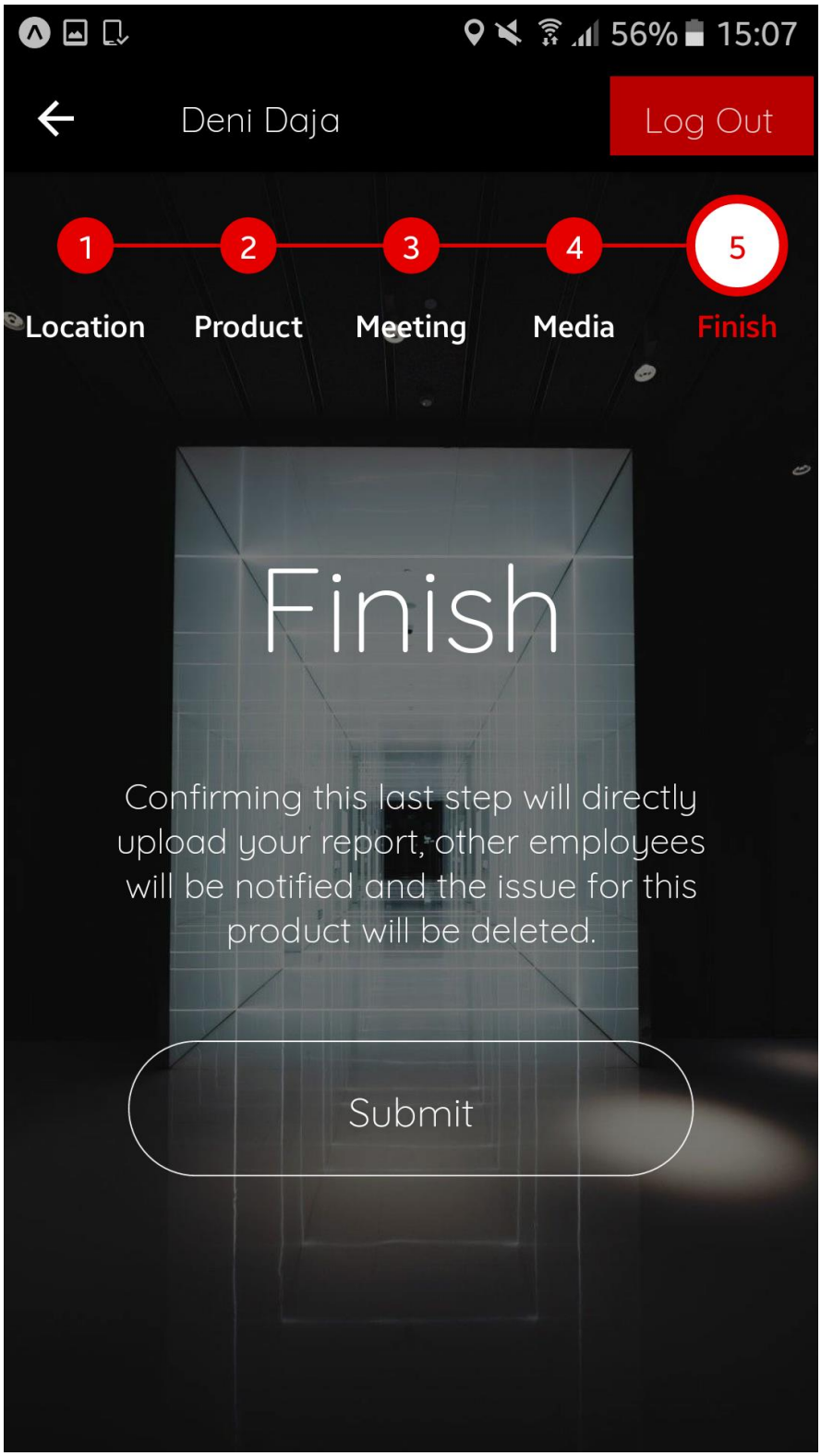



56% 15:07
Deni Daja
Log Out
1
2
3
4
5
Location
Product
Meeting
Media
Finish
Location
If you arrive at the location of the product please confirm and go to the next step. If you need directions for the location please refer to the first feature, click on the marker and watch the route.
Next 1/5


56% 15:07
Deni Daja
Log Out
1
2
3
4
5
Location
Product
Meeting
Media
Finish
Product
1. What issue did you encounter?
Description...
2. How did you deal with the issue?
Description...
Next 2/5


56% 15:07
Deni Daja
Log Out
1
2
3
4
5
Location
Product
Meeting
Media
Finish
Meeting
1. Who did you meet to discuss the issue?
Description...
2. What was the point of the discussion?
Description...
Next 3/5

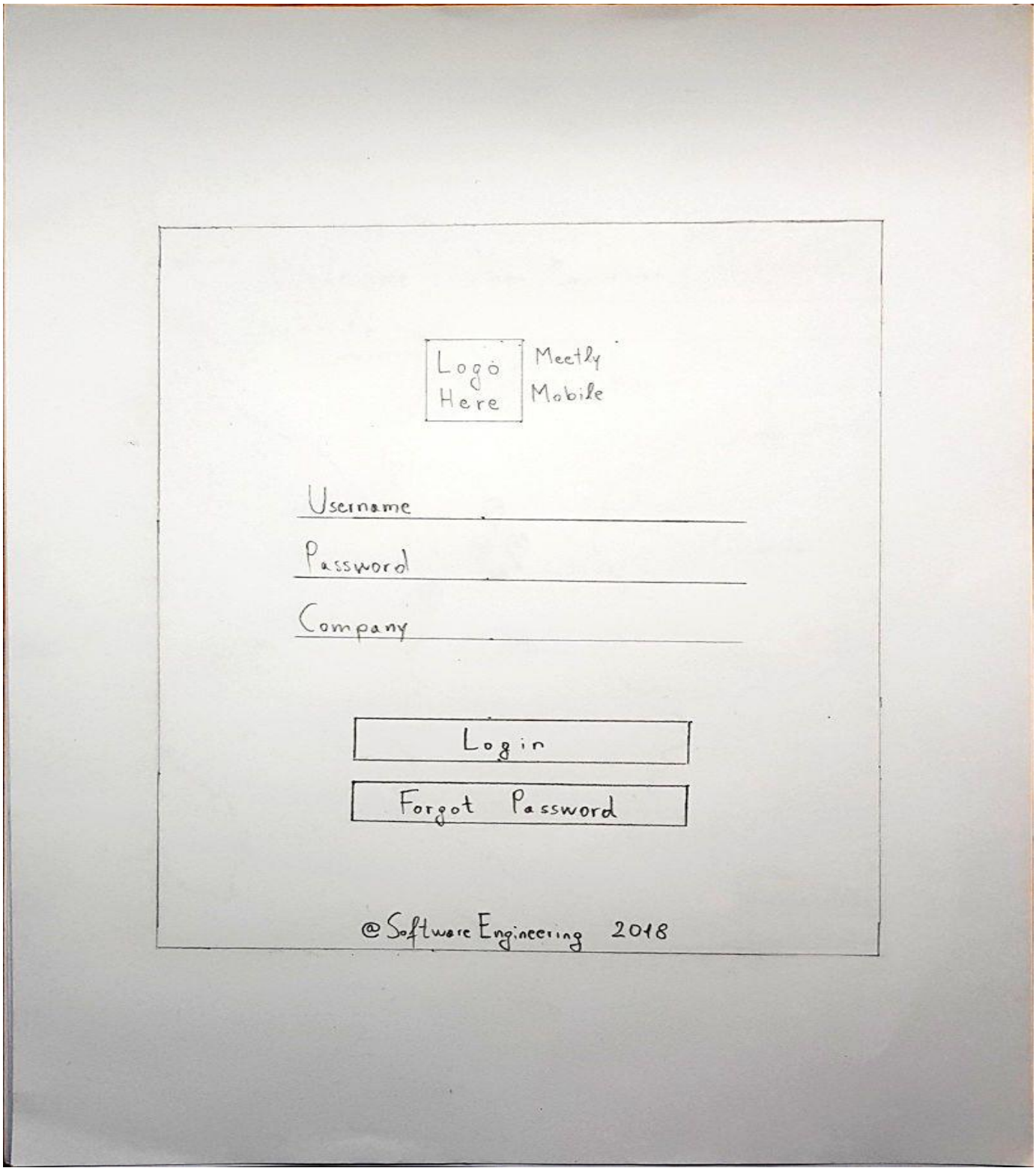

56% 15:07
Deni Daja
Log Out
1
2
3
4
5
Location
Product
Meeting
Media
Finish
Before
+
After
+
Comments
Description...
Next 4/5


56% 15:07
Deni Daja
Log Out
1
2
3
4
5
Location
Product
Meeting
Media
Finish
Finish
Confirming this last step will directly upload your report, other employees will be notified and the issue for this product will be deleted.
Submit

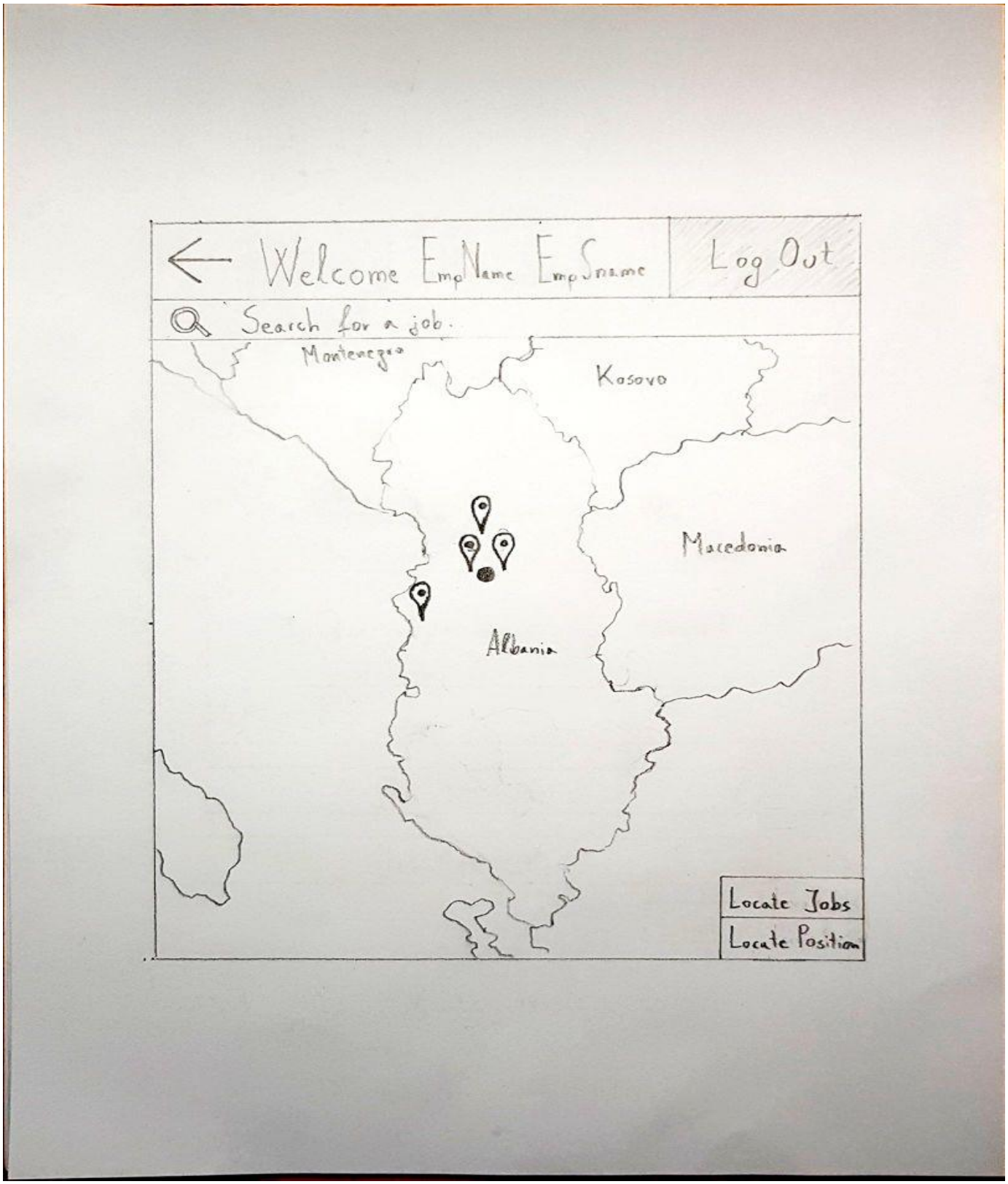
APPENDIX D

Sketches

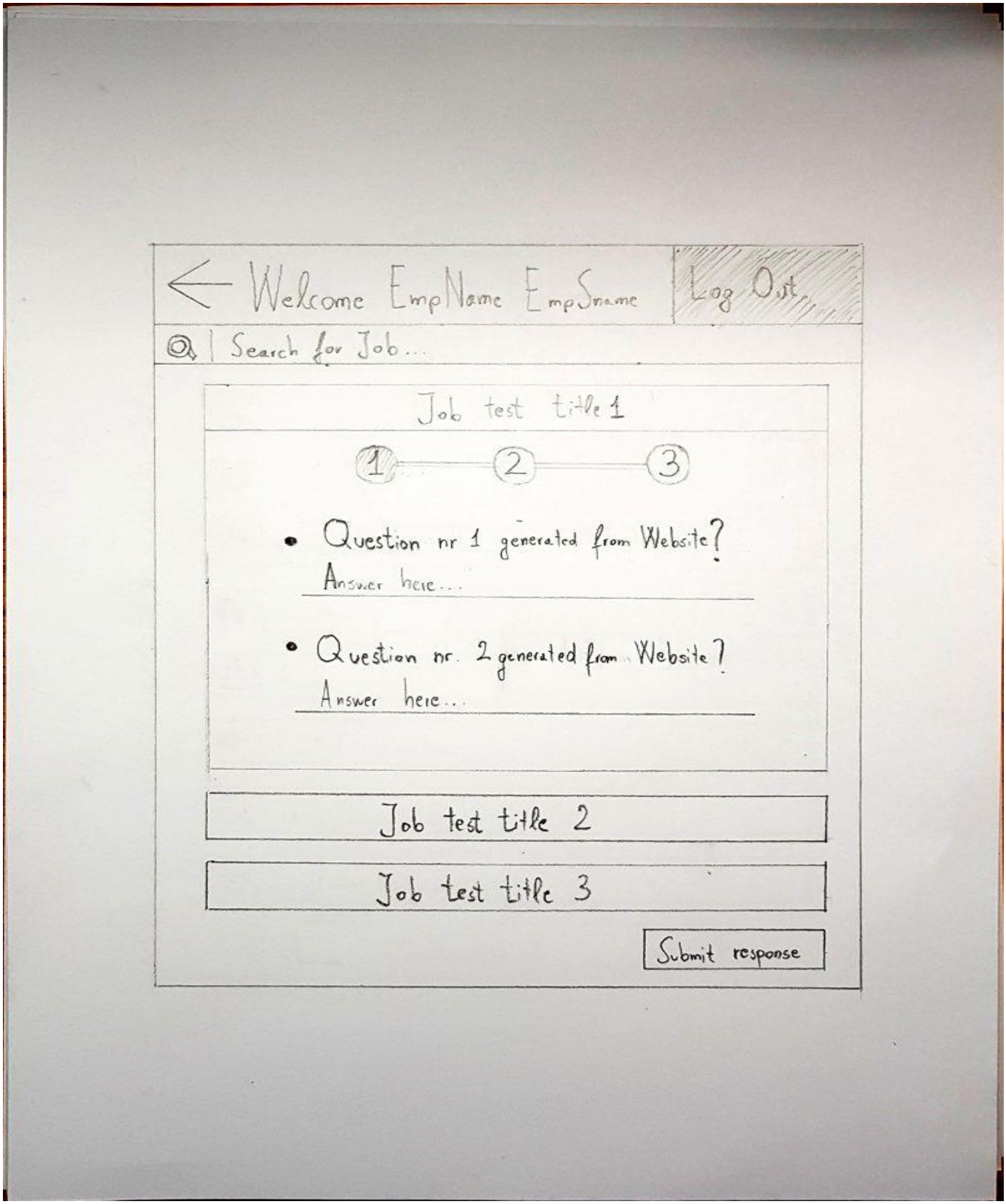
Meetly Mobile Application Login



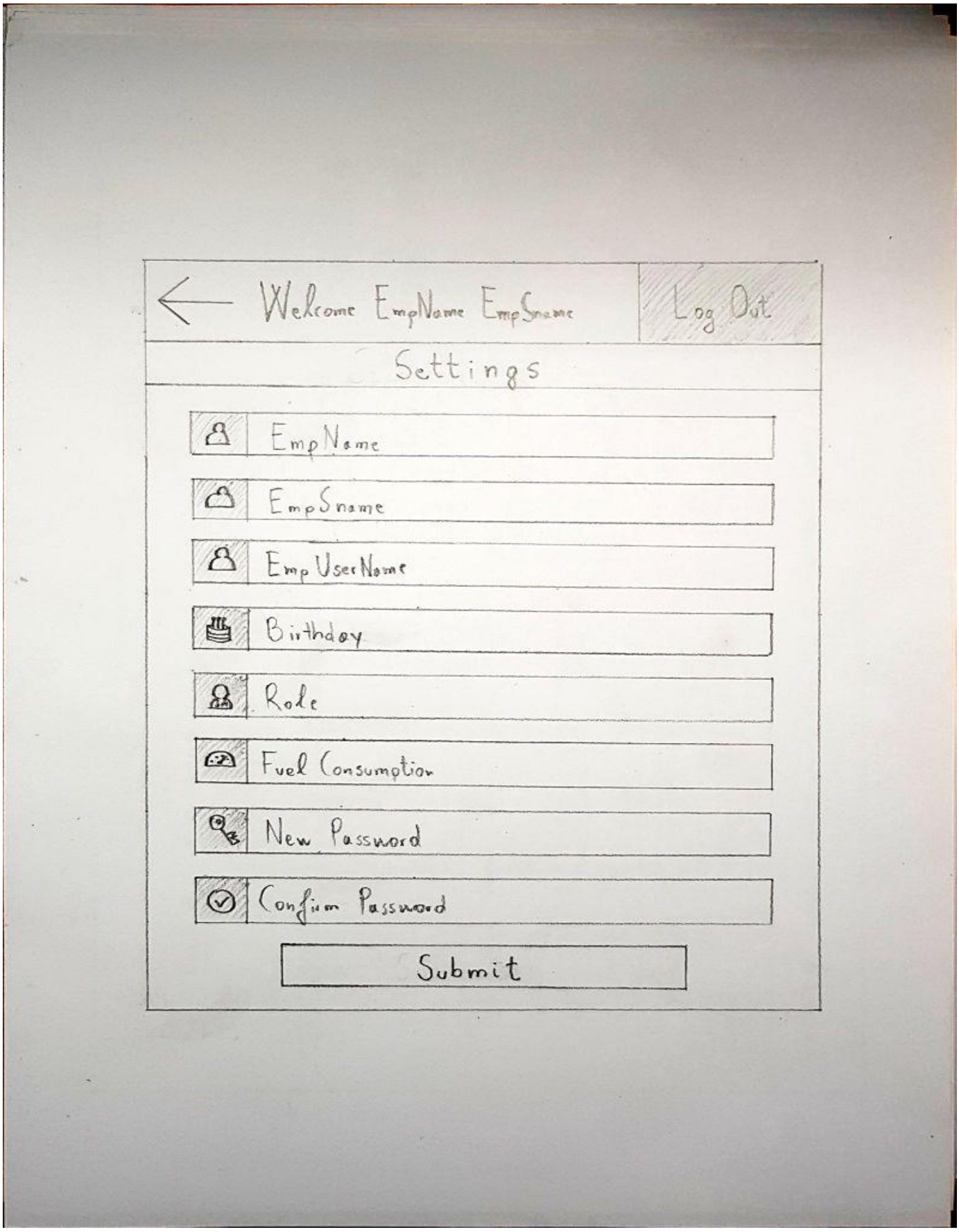
Meetly Mobile Application Map View



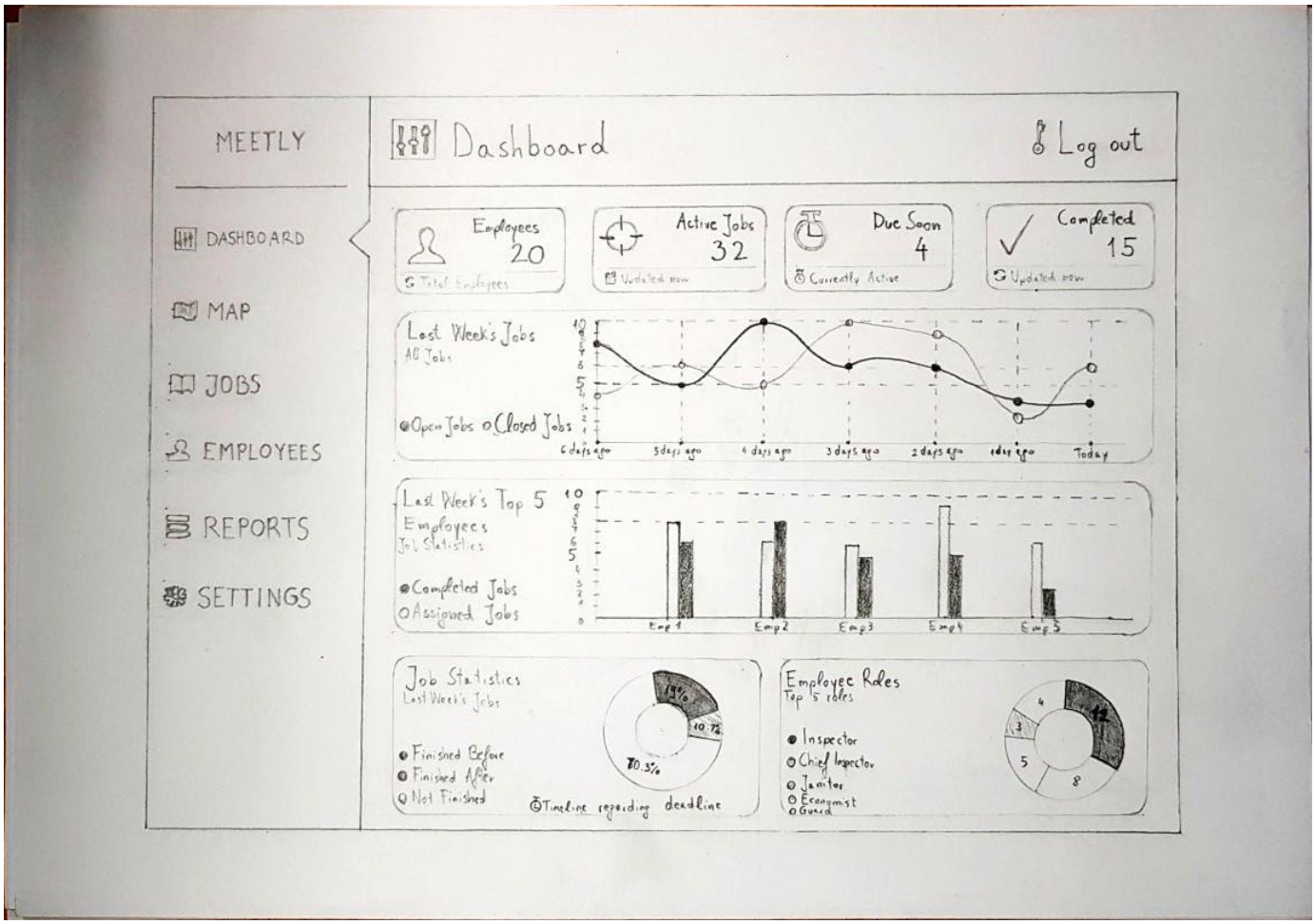
Meetly Mobile Application Job Response



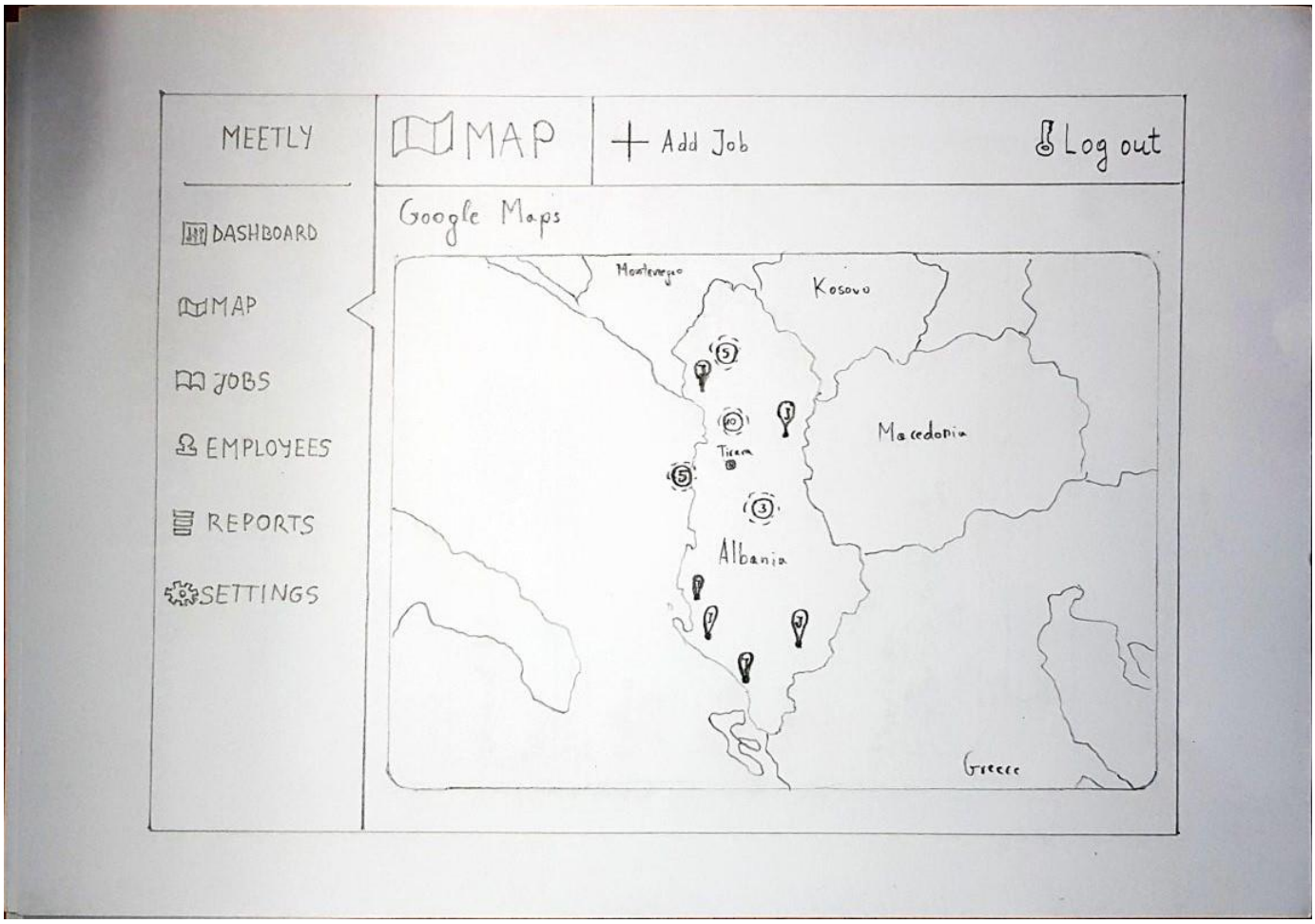
Meetly Mobile Application Edit Personal Information



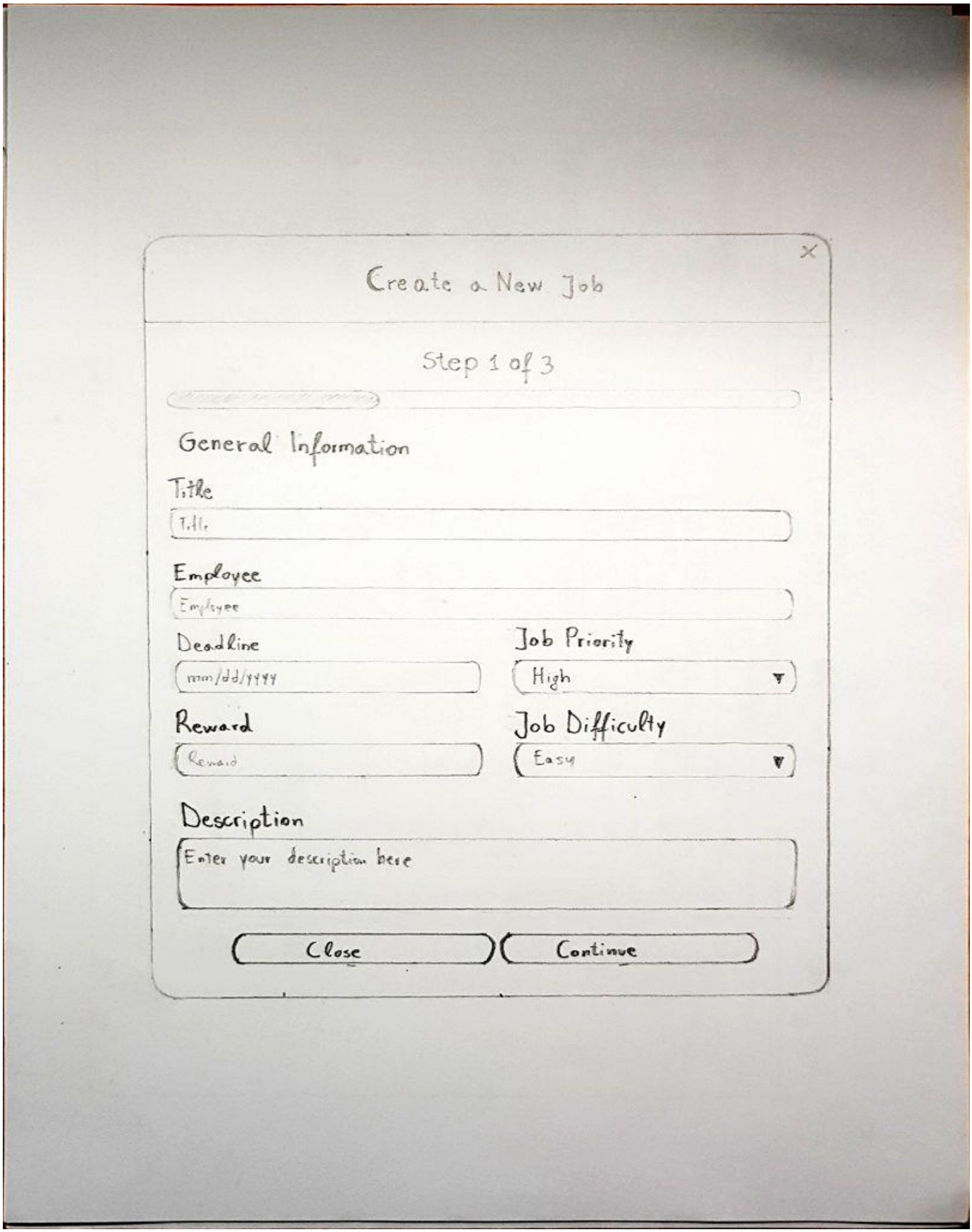
Meetly Web Dashboard



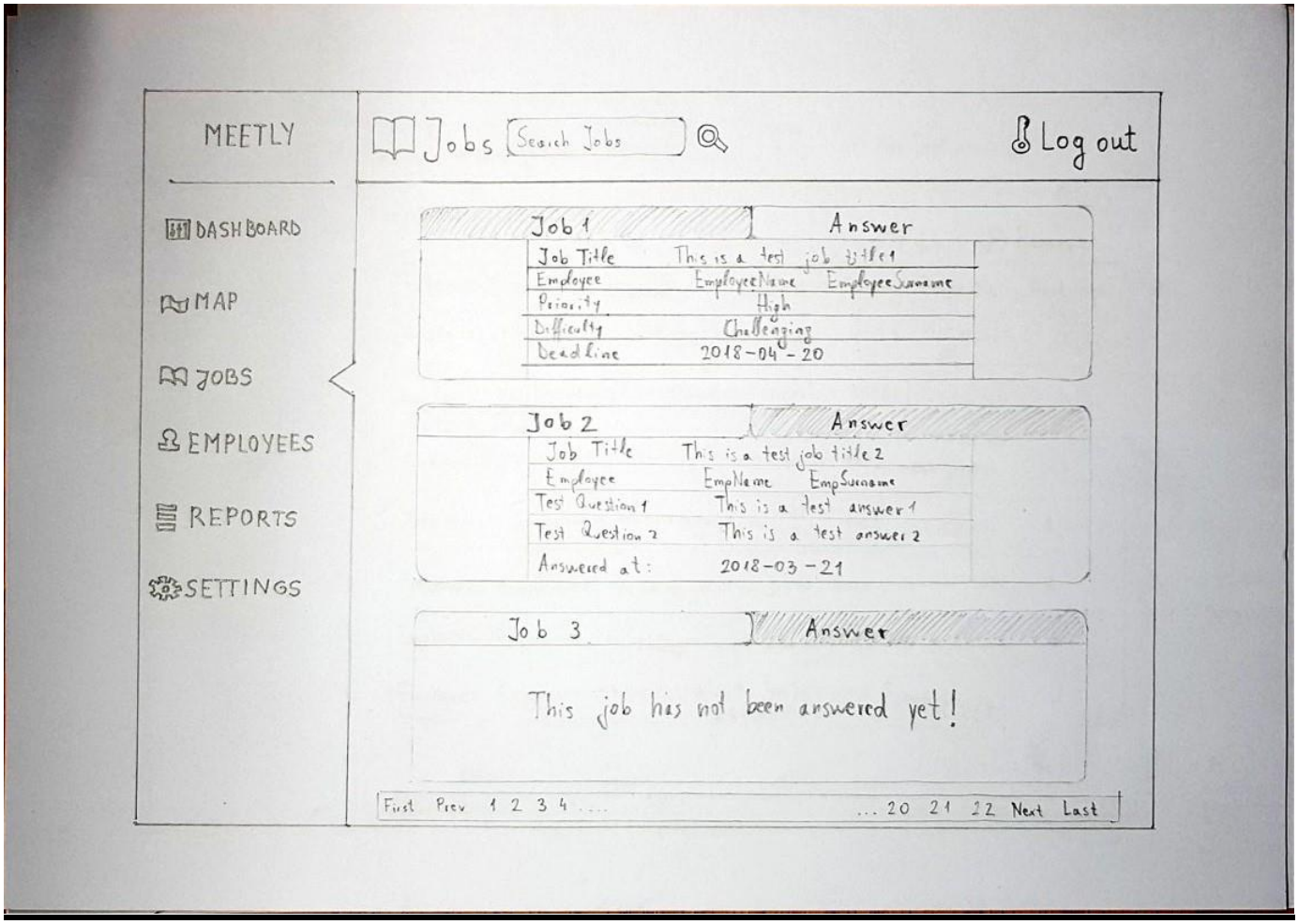
Meetly Web Map



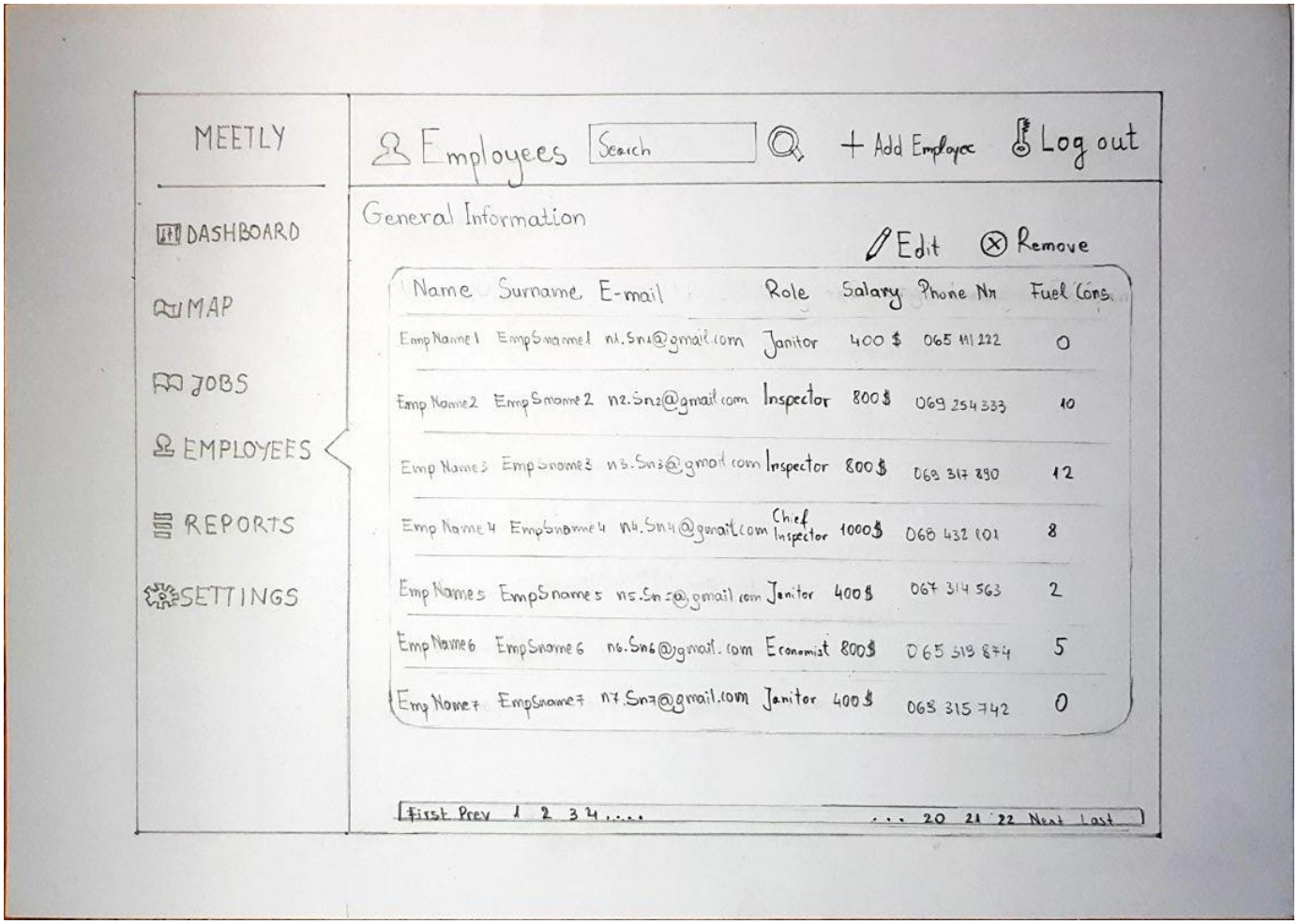
Meetly Web Create New Job Modal



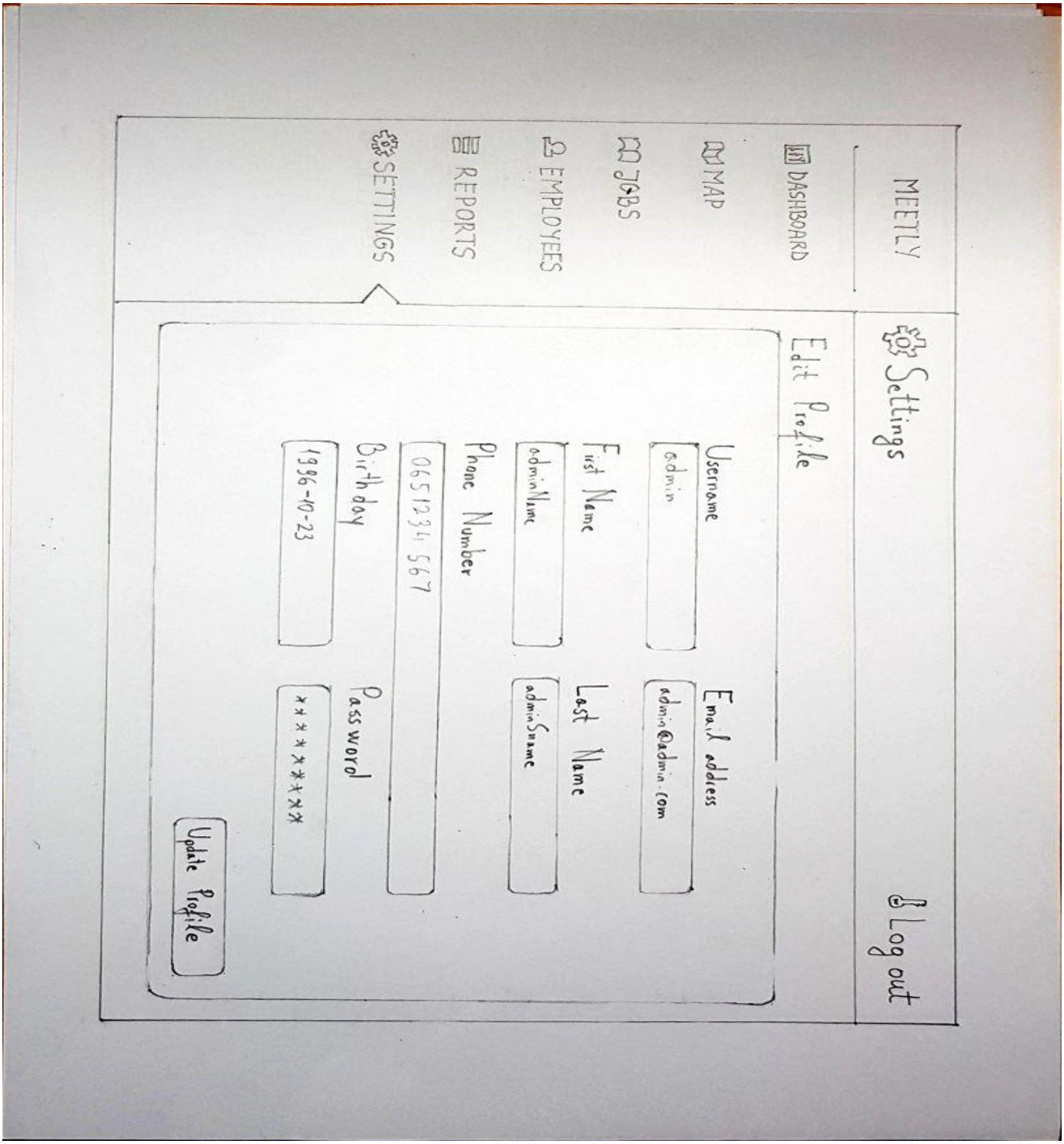
Meetly Web Jobs



Meetly Web Employees



Meetly Web Settings



APPENDIX E

Code

DefaultController.php

```
<?php

namespace AppBundle\Controller;

use AppBundle\Entity\Job;
use AppBundle\Entity\Notification;
use AppBundle\Entity\User;
use Faker\Provider\DateTime;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\HttpFoundation\Session\Session;

class DefaultController extends Controller
{
    /**
     * @Route("/", name="dashboard")
     */
    public function dashboardAction(Request $request)
    {

        $em = $this->getDoctrine()->getManager();

        // Employee Nr
        $sql = "SELECT COUNT(id) as employee_nr FROM user WHERE is_admin='0'";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $employee_nr = $stmt->fetchAll();

        // Open Jobs
        $sql = "SELECT COUNT(id) as open_jobs FROM job WHERE is_answered = '0'";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $open_jobs = $stmt->fetchAll();

        // Due Soon
        $sql = "SELECT COUNT(id) as due_soon FROM job WHERE deadline BETWEEN CURRENT_DATE() AND
(CURRENT_DATE() + INTERVAL 7 DAY)";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $due_soon = $stmt->fetchAll();

        // Closed Recently
        $sql = "SELECT COUNT(id) as recently FROM answer WHERE answered_at BETWEEN
(CURRENT_DATE() - INTERVAL 7 DAY) AND CURRENT_DATE()";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $recently = $stmt->fetchAll();

        // Last Week's opened and closed jobs
        $last_week_opened_jobs = array();
        $last_week_closed_jobs = array();

        for ($i = 6; $i >= 0; $i--) {
            $sql = "SELECT COUNT(id) as opened_jobs FROM job WHERE created_at = (CURRENT_DATE()
- INTERVAL $i DAY)";
            $stmt = $em->getConnection()->prepare($sql);
            $stmt->execute();
```

```
            $opened_jobs = $stmt->fetchAll();
            array_push($last_week_opened_jobs, $opened_jobs[0]['opened_jobs']);


            $sql = "SELECT COUNT(id) as closed_jobs FROM answer WHERE answered_at =
(CURRENT_DATE() - INTERVAL $i DAY)";
            $stmt = $em->getConnection()->prepare($sql);
            $stmt->execute();
            $closed_jobs = $stmt->fetchAll();
            array_push($last_week_closed_jobs, $closed_jobs[0]['closed_jobs']);
        }


        // Finished before deadline
        $sql = "SELECT COUNT(job.id) as finishedBeforeDeadline FROM job,answer WHERE
job.id=answer.job_id and answer.response is not null and job.deadline>=answer.answered_at";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $finishedBeforeDeadline = $stmt->fetchAll();

        // Finished after deadline
        $sql = "SELECT COUNT(job.id) as finishedAfterDeadline FROM job,answer WHERE
job.id=answer.job_id and answer.response is not null and job.deadline<=answer.answered_at";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $finishedAfterDeadline = $stmt->fetchAll();

        // Not finished at all
        $sql = "SELECT COUNT(id) as notFinished FROM job WHERE job.is_answered = 0";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $notFinished = $stmt->fetchAll();


        // Second Chart Query

        $sql = "SELECT count(job.id) as closed_jobs, job.user_id as robi, user.name,
user.surname, answered_at, (SELECT count(job.id)

FROM job

WHERE job.user_id = robi


) as assigned_jobs
                FROM job, user, answer
                WHERE job.user_id = user.id
                AND answer.job_id = job.id
                AND job.is_answered = 1
                AND answer.answered_at BETWEEN (CURRENT_DATE() - INTERVAL 7 DAY) AND
CURRENT_DATE()
                GROUP BY user_id
                ORDER BY count(job.id)
                DESC LIMIT 5;";

        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $top_employees = $stmt->fetchAll();


        // Fourth chart data, Roles and role_count

        $sql = "SELECT count(user.id) as role_count, role
                FROM user
                GROUP BY role
                ORDER BY count(user.id)
                DESC LIMIT 5";
```

```
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $top_roles = $stmt->fetchAll();

        $colors = array('#68B3C8', '#F3BB45', '#EB5E28', '#7AC29A', '#7A9E9F', 'rgba(104, 179,
200, 0.8)', 'rgba(122, 194, 154, 0.8)');

        $top_roles_colors = array();

        $cnt = 0;
        foreach ($top_roles as $role) {
            $role['color'] = $colors[$cnt++];
            array_push($top_roles_colors, $role);
        }


        // replace this example code with whatever you need
        return $this->render('Pages/dashboard.html.twig', [
            'user' => $this->getUser(),
            'stats' => [
                'employee_nr' => $employee_nr[0]['employee_nr'],
                'open_jobs' => $open_jobs[0]['open_jobs'],
                'due_soon' => $due_soon[0]['due_soon'],
                'recently' => $recently[0]['recently'],
                'last_week_opened_jobs' => $last_week_opened_jobs,
                'last_week_closed_jobs' => $last_week_closed_jobs,
                'finishedBeforeDeadline' =>
$finishedBeforeDeadline[0]['finishedBeforeDeadline'],
                'finishedAfterDeadline' => $finishedAfterDeadline[0]['finishedAfterDeadline'],
                'notFinished' => $notFinished[0]['notFinished'],
                'top_employees' => $top_employees,
                'top_roles' => $top_roles,
                'top_roles_colors' => $top_roles_colors
            ]
        ]);
    }

    /**
     * @Route("/paper", name="Paper")
     */
    public function paperAction(Request $request)
    {

        return $this->render('Pages/now-ui.html.twig', [

        ]);
    }



    /**
     * @Route("/maps", name="maps")
     */
    public function mapsAction(Request $request)
    {

        $em = $this->getDoctrine()->getManager();
        $sql = "select * from user where is_admin='0'";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $users = $stmt->fetchAll();

        $sql2 = "SELECT * FROM job WHERE is_answered = 0";
        $stmt2 = $em->getConnection()->prepare($sql2);
        $stmt2->execute();
        $jobs = $stmt2->fetchAll();
        // replace this example code with whatever you need
```

```
        return $this->render('Pages/maps.html.twig', [
            'users' => $users,
            'jobs' => $jobs,
        ]);
    }

    /**
     * @Route("/user", name="user")
     */
    public function userAction(Request $request)
    {
        // replace this example code with whatever you need
        return $this->render('Pages/user.html.twig', [
            'user' => $this->getUser()
        ]);
    }

    /**
     * @Route("/employee", name="employee")
     */
    public function employeeAction(Request $request)
    {
        // replace this example code with whatever you need
        $em = $this->getDoctrine()->getManager();
        $users = $em->getRepository('AppBundle:User')->findAll();
        return $this->render('Pages/employee.html.twig', [
            'users' => $users, 'num2' => 1, 'status' => 0
        ]);
    }

    /**
     * @Route("/employee/{slug}", name="employeepage")
     */
    public function employeeNewAction(Request $request, $slug)
    {
        // replace this example code with whatever you need
        $em = $this->getDoctrine()->getManager();
        $users = $em->getRepository('AppBundle:User')->findAll();
        return $this->render('Pages/employee.html.twig', [
            'users' => $users, 'num2' => $slug, 'status' => 0
        ]);
    }

    /**
     * @Route("/add/job",name="addJob")
     */
    public function addJobAction(Request $request)
    {

        $job = new Job();
        $job->setTitle($request->request->get('job-title'));
        $job->setDescription($request->request->get('job-desc'));
        $job->setForm($request->request->get('json-form'));
        $job->setLat($request->request->get('latitude'));
        $job->setLng($request->request->get('longitude'));

        $deadline = date_create($request->request->get('job-deadline'));

        $job->setDeadline($deadline);
        $job->setPriority($request->request->get('job-priority'));
        $job->setReward($request->request->get('job-reward'));
        $job->setJobDiff($request->request->get('job-difficulty'));
        $job->setIsAnswered(0);

        $employee = $request->request->get('job-employee');
        $data = explode(" ", $employee);
        $emp_name = $data[0];
```

```
        $emp_surname = $data[1];

        $user = $this->getDoctrine()->getRepository('AppBundle:User')->findOneBy([
            'name' => $emp_name,
            'surname' => $emp_surname
        ]);

        $job->setUser($user);

        $job->setUser($user);

        $date = new \DateTime();

        $job->setCreatedAt($date);
        $job->setUpdatedAt($date);

//        Get user name/surname from form and query for the user from the db
//        $user = new User();

        $em = $this->getDoctrine()->getManager();
        $em->persist($job);


        $notification = new Notification();

        $content = "New Job created by " . $this->getUser()->getName() . " " . $this-
>getUser()->getSurname() .
            ". The job is assigned to " . $user->getName() . " " . $user->getSurname() .
            ". Deadline: " . $request->request->get('job-deadline');

//        $date = new DateTime();

        $notification->setContent($content);
        $notification->setCreatedAt(new \DateTime());

        $em->persist($notification);
        $em->flush();

        return new Response('U kry');
    }


    /**
     * @Route("/delete/job",name="deleteJob")
     */
    public function deleteJobAction(Request $request)
    {

        $delete_id = $request->request->get('id');
        $em = $this->getDoctrine()->getManager();
        $sql = "DELETE FROM job WHERE id = $delete_id";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();

        return new Response('U eleminu');
    }

    /**
     * @Route("/get/employee",name="getEmployee")
     */
    public function getEmployeeAction(Request $request)
    {

        $emp_id = $request->request->get('id');
        $em = $this->getDoctrine()->getManager();
        $sql = "select * from user wh ere id = $emp_id";
        $stmt = $em->getConnection()->prepare($sql);
```

```
        $stmt->execute();
        $user = $stmt->fetchAll();

        return new Response($user[0]['name'] . " " . $user[0]['surname']);
    }

    /**
     * @Route("/get/notification",name="getNotification")
     */
    public function getNotificationAction(Request $request)
    {
        $em = $this->getDoctrine()->getManager();

        $sql = "SELECT * FROM notification WHERE created_at >= DATE_SUB(NOW(),INTERVAL 5
HOUR)";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $notification = $stmt->fetchAll();

        return new Response(json_encode($notification));
    }


    /**
     * @Route("/jobs",name="jobs")
     */
    public function jobsAction(Request $request)
    {

        $em = $this->getDoctrine()->getManager();

        $sql = "SELECT job.id as job_id, user_id, title, deadline, description, form, lat, lng,
                    deadline, priority, reward, job_diff, job.created_at, job.updated_at,
is_answered,
                    a.id as answer_id, a.response, answered_at, u.name, u.surname
                FROM job
                JOIN user u ON job.user_id = u.id
                LEFT JOIN answer a ON job.id = a.job_id ORDER BY job.created_at DESC";
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $jobs = $stmt->fetchAll();


        return $this->render('Pages/jobs.html.twig', [
            'jobs' => $jobs,
            'response' => $jobs[0]['response']
        ]);
    }

    /**
     * @Route("/search/jobs",name="searchJobs")
     */
    public function searchJobsAction(Request $request)
    {

        $searching = $request->request->get('job-title');

        $em = $this->getDoctrine()->getManager();

        $sql = "SELECT job.id as job_id, user_id, title, deadline, description, form, lat, lng,
                    deadline, priority, reward, job_diff, job.created_at, job.updated_at,
is_answered,
                    a.id as answer_id, a.response, answered_at, u.name, u.surname
                FROM job
                JOIN user u ON job.user_id = u.id
                LEFT JOIN answer a ON job.id = a.job_id
                WHERE job.title LIKE '%$searching%'
```

```
                ORDER BY job.created_at DESC";

        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $jobs = $stmt->fetchAll();

        return new Response(json_encode($jobs));
    }


    /**
     * @Route("/delete/jobs",name="deleteJobs")
     */
    public function deleteJobsAction(Request $request)
    {

        $job_id = $request->request->get('job-id');

        $em = $this->getDoctrine()->getManager();
        $sql = "DELETE FROM job WHERE job.id = $job_id";

        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();

        $answer_id = $request->request->get('answer_id');

        if ($answer_id != null) {
            $sql = "DELETE FROM answer WHERE answer.id = $answer_id AND answer.job_id =
$job_id";
            $stmt = $em->getConnection()->prepare($sql);
            $stmt->execute();
        }


        return new Response("yes");
    }

    /**
     * @Route("/add",name="addAction")
     */

    public function addAction()
    {
        return $this->render("/Pages/add.html.twig", ['status' => 3]);
    }

    /**
     * @Route("/create",name="cremp")
     */
    public function create_new_emp(Request $request)
    {

        $em = $this->getDoctrine()->getManager();
        $username = $request->request->get('text-username');
        $email = $request->request->get('text-email');
        $name = $request->request->get("text-name");
        $surname = $request->request->get('text-surname');
        $oldate = $request->request->get('text-birthday');
        $newdate = date('Y-m-d', strtotime($oldate));
        $role = $request->request->get('text-role');
        $salary = $request->request->get('text-salary');
        $password = $request->request->get('text-password');
        $conf_passwor = $request->request->get('text-password-conf');
        $admin = $request->request->get('text-admin');
        $phone = $request->request->get('text-phone');

        if ($password == $conf_passwor) {
```

```
            $sql = "insert into user
(username,password,name,surname,birthday,email,phone_nr,is_admin,role,salary,created_at,updated
_at,last_loggin,fuel_consumption)
            values
('$username','$password','$name','$surname','$newdate','$email','$phone',$admin,'.$role.',$sala
ry,000-00-00,000-00-00,000-00-00,0.0)";
            $stmt = $em->getConnection()->prepare($sql);
            $stmt->execute();
            return $this->render("/Pages/add.html.twig", ['status' => 1]);
        } else {
            return $this->render("/Pages/add.html.twig", ['status' => 0]);
        }

    }

    /**
     * @Route("/edit/{slug}",name="editemp")
     */
    public function edit_employee($slug)
    {
        $em = $this->getDoctrine()->getManager();
        $sql = "select * from user where id =" . $slug;
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $users = $stmt->fetchAll();
        return $this->render('/Pages/edit.html.twig', ['users' => $users]);
    }

    /**
     * @Route("/edit/conf",name="config")\
     */
    public function edit_employee_confirmation(Request $request)
    {
        $em = $this->getDoctrine()->getManager();
        $username = $request->request->get('text-username');
        $email = $request->request->get('text-email');
        $phone = $request->request->get('text-phone');
        $role = $request->request->get('text-role');
        $salary = $request->request->get('text-salary');
        $password = $request->request->get('text-password');
        $pass_conf = $request->request->get('text-password-confirm');
        $id = $request->request->get('text-id');

        if ($password == $pass_conf) {
            $sql = "update user set user.username = $username, user.email = $email,
user.phone_nr = $phone, user.role = $role, user.salary = $salary, user.password = $password
             where user.id =" . $id;
            $stmt = $em->getConnection()->prepare($sql);
            $stmt->execute();
            return $this->redirectToRoute('employee');
        } else {
            $sql = "select * from user where id =" . $id;
            $stmt = $em->getConnection()->prepare($sql);
            $stmt->execute();
            $users = $stmt->fetchAll();
            $stmt->close();

            return $this->render('/Pages/edit.html.twig', ['users' => $users, 'status' => 0]);
        }

    }

    /**
     * @Route("/remove/{slug}",name="rem")
     */
    public function removeEmployee($slug)
    {
```

```
        $em = $this->getDoctrine()->getManager();
        $sql = "delete from user where user.id=" . $slug;
        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        return $this->redirectToRoute('employee');

    }

    /**
     * @Route("search",name="search")
     */
    public function searchEmployee(Request $request)
    {
        $em = $this->getDoctrine()->getManager();
        $text = $request->request->get('text');
        $sql = "SELECT * from user WHERE name LIKE '%" . $text . "%' OR email LIKE '%" . $text
. "%' OR surname LIKE '%" . $text . "%'";

        $stmt = $em->getConnection()->prepare($sql);
        $stmt->execute();
        $user = $stmt->fetchAll();

        return $this->render("/Pages/employee.html.twig", ['users' => $user, 'num2' => 1,
'status' => 1]);
    }

    /**
     * @Route("/reports", name="reports")
     */
    public function reportsAction(Request $request)
    {
        // replace this example code with whatever you need
        $em = $this->getDoctrine()->getManager();
        $sql_user = "select * from user where user.is_admin=0";
        $sql_chief = "select * from user where user.is_admin=1";
        $stmt1 = $em->getConnection()->prepare($sql_user);
        $stmt1->execute();
        $users = $stmt1->fetchAll();
        $stmt2 = $em->getConnection()->prepare($sql_chief);
        $stmt2->execute();
        $chiefs = $stmt2->fetchAll();

        return $this->render('Pages/reports.html.twig', [
            'users' => $users, 'chiefs' => $chiefs
        ]);
    }
}
```

Login Controller

```php
<?php

namespace AppBundle\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\Controller;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Security\Http\Authentication\AuthenticationUtils;


class LoginController extends Controller
{
    /**
     * @Route("/login",name="login")
     */
    public function loginAction(Request $request,AuthenticationUtils $authenticationUtils)
    {

        $errors = $authenticationUtils->getLastAuthenticationError();

        $lastUsername = $authenticationUtils->getLastUsername();

        return $this->render("Login/login.html.twig",array(
            'errors'=>$errors,
            'username'=>$lastUsername
        ));
    }

    /**
     * @Route("/logout",name="logout")
     */
    public function logoutAction()
    {

    }

}
```

User.php

```php
<?php
/**
 * Created by PhpStorm.
 * User: User
 * Date: 3/8/2018
 * Time: 6:16 PM
 */

namespace AppBundle\Entity;


use Doctrine\ORM\Mapping as ORM;
use Faker\Provider\Base;
use Symfony\Component\Security\Core\User\UserInterface;
/**
 * @ORM\Entity
 * @ORM\Table(name="user")
 */
class User implements UserInterface
{

    /**
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="AUTO")
     * @ORM\Column(type="integer")
     */
    private $id;


    /**
     * @ORM\Column(type="string")
     */
    private $username;


    /**
     * @ORM\Column(type="string")
     */
    private $password;


    /**
     * @ORM\Column(type="string")
     */
    private $name;


    /**
     * @ORM\Column(type="string")
     */
    private $surname;


    /**
     * @ORM\Column(type="date")
     */
    private $birthday;


    /**
     * @ORM\Column(type="string")
     */
    private $email;
```

```php
    /**
     * @ORM\Column(type="string")
     */
    private $phone_nr;


    /**
     * @ORM\Column(type="boolean")
     */
    private $is_admin;


    /**
     * @ORM\Column(type="string")
     */
    private $role;


    /**
     * @ORM\Column(type="float")
     */
    private $salary;


    /**
     * @ORM\Column(type="date")
     */
    private $created_at;


    /**
     * @ORM\Column(type="date")
     */
    private $updated_at;


    /**
     * @ORM\Column(type="datetime")
     */
    private $last_loggin;


    /**
     * @ORM\Column(type="float")
     */
    private $fuel_consumption;

    /**
     * @return mixed
     */
    public function getId()
    {
        return $this->id;
    }

    /**
     * @param mixed $id
     */
    public function setId($id)
    {
        $this->id = $id;
    }

    /**
     * @return mixed
```

```
     */
    public function getUsername()
    {
        return $this->username;
    }

    /**
     * @param mixed $username
     */
    public function setUsername($username)
    {
        $this->username = $username;
    }

    /**
     * @return mixed
     */
    public function getPassword()
    {
        return $this->password;
    }

    /**
     * @param mixed $password
     */
    public function setPassword($password)
    {
        $this->password = $password;
    }

    /**
     * @return mixed
     */
    public function getName()
    {
        return $this->name;
    }

    /**
     * @param mixed $name
     */
    public function setName($name)
    {
        $this->name = $name;
    }

    /**
     * @return mixed
     */
    public function getSurname()
    {
        return $this->surname;
    }

    /**
     * @param mixed $surname
     */
    public function setSurname($surname)
    {
        $this->surname = $surname;
    }

    /**
     * @return mixed
     */
    public function getBirthday()
    {
```

```
        return $this->birthday;
    }

    /**
     * @param mixed $birthday
     */
    public function setBirthday($birthday)
    {
        $this->birthday = $birthday;
    }

    /**
     * @return mixed
     */
    public function getEmail()
    {
        return $this->email;
    }

    /**
     * @param mixed $email
     */
    public function setEmail($email)
    {
        $this->email = $email;
    }

    /**
     * @return mixed
     */
    public function getPhoneNr()
    {
        return $this->phone_nr;
    }

    /**
     * @param mixed $phone_nr
     */
    public function setPhoneNr($phone_nr)
    {
        $this->phone_nr = $phone_nr;
    }

    /**
     * @return mixed
     */
    public function getIsAdmin()
    {
        return $this->is_admin;
    }

    /**
     * @param mixed $is_admin
     */
    public function setIsAdmin($is_admin)
    {
        $this->is_admin = $is_admin;
    }

    /**
     * @return mixed
     */
    public function getRole()
    {
        return $this->role;
    }
```

```php
    /**
     * @param mixed $role
     */
    public function setRole($role)
    {
        $this->role = $role;
    }

    /**
     * @return mixed
     */
    public function getSalary()
    {
        return $this->salary;
    }

    /**
     * @param mixed $salary
     */
    public function setSalary($salary)
    {
        $this->salary = $salary;
    }

    /**
     * @return mixed
     */
    public function getCreatedAt()
    {
        return $this->created_at;
    }

    /**
     * @param mixed $created_at
     */
    public function setCreatedAt($created_at)
    {
        $this->created_at = $created_at;
    }

    /**
     * @return mixed
     */
    public function getUpdatedAt()
    {
        return $this->updated_at;
    }

    /**
     * @param mixed $updated_at
     */
    public function setUpdatedAt($updated_at)
    {
        $this->updated_at = $updated_at;
    }

    /**
     * @return mixed
     */
    public function getLastLoggin()
    {
        return $this->last_loggin;
    }

    /**
     * @param mixed $last_loggin
     */
```

```
    public function setLastLoggin($last_loggin)
    {
        $this->last_loggin = $last_loggin;
    }

    /**
     * @return mixed
     */
    public function getFuelConsumption()
    {
        return $this->fuel_consumption;
    }

    /**
     * @param mixed $fuel_consumption
     */
    public function setFuelConsumption($fuel_consumption)
    {
        $this->fuel_consumption = $fuel_consumption;
    }


    /**
     * Returns the roles granted to the user.
     *
     * <code>
     * public function getRoles()
     * {
     *     return array('ROLE_USER');
     * }
     * </code>
     *
     * Alternatively, the roles might be stored on a ``roles`` property,
     * and populated in any number of different ways when the user object
     * is created.
     *
     * @return (Role|string)[] The user roles
     */
    public function getRoles()
    {
        // TODO: Implement getRoles() method.
        return [
            'ROLE_USER'
        ];
    }

    /**
     * Returns the salt that was originally used to encode the password.
     *
     * This can return null if the password was not encoded using a salt.
     *
     * @return string|null The salt
     */
    public function getSalt()
    {
        // TODO: Implement getSalt() method.
    }

    /**
     * Removes sensitive data from the user.
     *
     * This is important if, at any given point, sensitive information like
     * the plain-text password is stored on this object.
     */
    public function eraseCredentials()
    {
        // TODO: Implement eraseCredentials() method.
```

```
    }
}
```

Answer.php

```php
<?php
/**
 * Created by PhpStorm.
 * User: Admin
 * Date: 03/13/2018
 * Time: 6:18 PM
 */

namespace AppBundle\Entity;


use Doctrine\ORM\Mapping as ORM;

/**
 * @ORM\Entity
 * @ORM\Table(name="answer")
 */
class Answer
{
    /**
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="AUTO")
     * @ORM\Column(type="integer")
     */
    private $id;


    /**
     * @ORM\OneToOne(targetEntity="AppBundle\Entity\Job")
     * @ORM\JoinColumn(nullable=false)
     */
    private $job;


    /**
     * @ORM\Column(type="text")
     */
    private $response;


    /**
     * @ORM\Column(type="date")
     */
    private $answered_at;

    /**
     * @return mixed
     */
    public function getId()
    {
        return $this->id;
    }

    /**
     * @param mixed $id
     */
    public function setId($id)
    {
        $this->id = $id;
```

```
    }

    /**
     * @return mixed
     */
    public function getJob()
    {
        return $this->job;
    }

    /**
     * @param mixed $job
     */
    public function setJob($job)
    {
        $this->job = $job;
    }

    /**
     * @return mixed
     */
    public function getResponse()
    {
        return $this->response;
    }

    /**
     * @param mixed $response
     */
    public function setResponse($response)
    {
        $this->response = $response;
    }

    /**
     * @return mixed
     */
    public function getAnsweredAt()
    {
        return $this->answered_at;
    }

    /**
     * @param mixed $answered_at
     */
    public function setAnsweredAt($answered_at)
    {
        $this->answered_at = $answered_at;
    }



}
```

Job.php

```
<?php
/**
 * Created by PhpStorm.
 * User: User
 * Date: 3/8/2018
 * Time: 6:26 PM
 */
```

```php
namespace AppBundle\Entity;

use Doctrine\ORM\Mapping as ORM;

/**
 * @ORM\Entity
 * @ORM\Table(name="job")
 */
class Job
{

    /**
     * @ORM\Id
     * @ORM\GeneratedValue(strategy="AUTO")
     * @ORM\Column(type="integer")
     */private $id;


    /**
     * @ORM\ManyToOne(targetEntity="User")
     * @ORM\JoinColumn(onDelete="SET NULL")
     */
    private $user;


    /**
     * @ORM\Column(type="string")
     */
    private $title;


    /**
     * @ORM\Column(type="string")
     */
    private $description;


    /**
     * @ORM\Column(type="text")
     */
    private $form;


    /**
     * @ORM\Column(type="string")
     */
    private $lat;


    /**
     * @ORM\Column(type="string")
     */
    private $lng;


    /**
     * @ORM\Column(type="date")
     */
    private $deadline;


    /**
     * @ORM\Column(type="integer")
     */
    private $priority;
```

```
    /**
     * @ORM\Column(type="float")
     */
    private $reward;


    /**
     * @ORM\Column(type="integer")
     */
    private $job_diff;


    /**
     * @ORM\Column(type="date")
     */
    private $created_at;


    /**
     * @ORM\Column(type="date")
     */
    private $updated_at;


    /**
     * @ORM\Column(type="boolean")
     */
    private $is_answered;


    /**
     * @return mixed
     */
    public function getId()
    {
        return $this->id;
    }

    /**
     * @param mixed $id
     */
    public function setId($id)
    {
        $this->id = $id;
    }

    /**
     * @return mixed
     */
    public function getUser()
    {
        return $this->user;
    }

    /**
     * @param mixed $user
     */
    public function setUser(User $user)
    {
        $this->user = $user;
    }

    /**
     * @return mixed
     */
    public function getTitle()
    {
```

```
        return $this->title;
    }

    /**
     * @param mixed $title
     */
    public function setTitle($title)
    {
        $this->title = $title;
    }

    /**
     * @return mixed
     */
    public function getDescription()
    {
        return $this->description;
    }

    /**
     * @param mixed $description
     */
    public function setDescription($description)
    {
        $this->description = $description;
    }

    /**
     * @return mixed
     */
    public function getForm()
    {
        return $this->form;
    }

    /**
     * @param mixed $form
     */
    public function setForm($form)
    {
        $this->form = $form;
    }

    /**
     * @return mixed
     */
    public function getLat()
    {
        return $this->lat;
    }

    /**
     * @param mixed $lat
     */
    public function setLat($lat)
    {
        $this->lat = $lat;
    }

    /**
     * @return mixed
     */
    public function getLng()
    {
        return $this->lng;
    }
```

```
    /**
     * @param mixed $lng
     */
    public function setLng($lng)
    {
        $this->lng = $lng;
    }

    /**
     * @return mixed
     */
    public function getDeadline()
    {
        return $this->deadline;
    }

    /**
     * @param mixed $deadline
     */
    public function setDeadline($deadline)
    {
        $this->deadline = $deadline;
    }

    /**
     * @return mixed
     */
    public function getPriority()
    {
        return $this->priority;
    }

    /**
     * @param mixed $priority
     */
    public function setPriority($priority)
    {
        $this->priority = $priority;
    }

    /**
     * @return mixed
     */
    public function getReward()
    {
        return $this->reward;
    }

    /**
     * @param mixed $reward
     */
    public function setReward($reward)
    {
        $this->reward = $reward;
    }

    /**
     * @return mixed
     */
    public function getJobDiff()
    {
        return $this->job_diff;
    }

    /**
     * @param mixed $job_diff
     */
```

```
    public function setJobDiff($job_diff)
    {
        $this->job_diff = $job_diff;
    }

    /**
     * @return mixed
     */
    public function getCreatedAt()
    {
        return $this->created_at;
    }

    /**
     * @param mixed $created_at
     */
    public function setCreatedAt($created_at)
    {
        $this->created_at = $created_at;
    }

    /**
     * @return mixed
     */
    public function getUpdatedAt()
    {
        return $this->updated_at;
    }

    /**
     * @param mixed $updated_at
     */
    public function setUpdatedAt($updated_at)
    {
        $this->updated_at = $updated_at;
    }

    /**
     * @return mixed
     */
    public function getIsAnswered()
    {
        return $this->is_answered;
    }

    /**
     * @param mixed $is_answered
     */
    public function setIsAnswered($is_answered)
    {
        $this->is_answered = $is_answered;
    }



}
```

Login.html.twig

```
{% extends 'base.html.twig' %}
{% block body %}

    {% block stylesheets %}
        <link href="https://fonts.googleapis.com/css?family=Montserrat:400,700,200"
rel="stylesheet"/>
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-
awesome/latest/css/font-awesome.min.css"/>
        <!-- CSS Files -->
        <link href="/loginassets/css/bootstrap.min.css" rel="stylesheet"/>
        <link href="/loginassets/css/now-ui-kit.css?v=1.1.0" rel="stylesheet"/>
    {% endblock %}

    <nav class="navbar navbar-expand-lg bg-primary fixed-top navbar-transparent " color-on-
scroll="400">
        <div class="container">
            <div class="collapse navbar-collapse justify-content-end" id="navigation"
                 data-nav-image="assets/img/bg.jpg">
                <ul class="navbar-nav">
                    <li class="nav-item">
                        <a class="nav-link" rel="tooltip" title="Follow us on Twitter" data-
placement="bottom" href=""
                           target="_blank">
                            <i class="fa fa-twitter"></i>
                            <p class="d-lg-none d-xl-none">Twitter</p>
                        </a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" rel="tooltip" title="Like us on Facebook" data-
placement="bottom" href=""
                           target="_blank">
                            <i class="fa fa-facebook-square"></i>
                            <p class="d-lg-none d-xl-none">Facebook</p>
                        </a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" rel="tooltip" title="Follow us on Instagram" data-
placement="bottom" href=""
                           target="_blank">
                            <i class="fa fa-instagram"></i>
                            <p class="d-lg-none d-xl-none">Instagram</p>
                        </a>
                    </li>
                </ul>
            </div>
        </div>
    </nav>
    <!-- End Navbar -->

    <div class="panel-body">

        <div class="page-header" filter-color="orange">
            <div class="page-header-image" style="background-
image:url('/loginassets/img/bg.jpg')"></div>
            <div class="container">
                <div class="col-md-4 content-center">
                    <center>
                        <div class="card card-login card-plain">
                            <div class="header header-primary text-center">
                                <div class="logo-container">
                                    <p style="font-size: 52px;font-weight:
100">Meet<strong><span
                                                        style="color: #e60000;font-size:
```

```
22px">LY</span></strong></p>
                                </div>
                            </div>

                            <form action="{{ path('login') }}" method="POST">
                                <div class="content" style="width: 50%">
                                    <div class="input-group form-group-no-border input-lg">
                                    <span class="input-group-addon">
                                        <i class="now-ui-icons users_circle-08" style="color:
white"></i>
                                    </span>
                                        <input placeholder="Username..." type="text"
name="_username" id="username"
                                               class="form-control" value="{{ username }}"
style="color: white;"/>
                                    </div>
                                    <div class="input-group form-group-no-border input-lg">
                                    <span class="input-group-addon">
                                        <i class="now-ui-icons text_caps-small" style="color:
white"></i>
                                    </span>
                                        <input placeholder="Password..." type="text"
name="_password" id="password"
                                               class="form-control" style="color: white;"/>
                                    </div>
                                </div>
                                <div class="footer text-center" style="width: 50%">
                                    <button class="btn btn-primary btn-round btn-lg btn-block"
type="submit"
                                            style="background-color: #e60000" id="sb">Login
                                    </button>
                                </div>
                            </form>

                            {#<div class="pull-center">#}
                            {#<h6>#}
                            {#<a class="link" style="color: white">Need Help?</a>#}
                            {#</h6>#}
                            {#</div>#}

                            {% if errors %}

                                <div class="alert alert-danger"
                                     style="border-radius: 20px;background-color:
transparent;width: 300px;font-size: 22px;color: white">
                                    {{ errors.messageKey }}
                                </div>

                            {% endif %}


                        </div>
                    </center>
                </div>
            </div>
            <footer class="footer">
                <div class="container">
                    <nav>
                        <ul>
                            <li>
                                <a href="#">
                                    Epoka University
                                </a>
                            </li>
                        </ul>
                    </nav>
                    <div class="copyright">
```

```
                        &copy;
                        <script>
                            document.write(new Date().getFullYear())
                        </script>
                        Coded by
                        <a target="_blank">Software Engineering</a>.
                    </div>
                </div>
            </footer>
        </div>

    </div>

{% endblock %}


{% block javascrips %}

    <!--   Core JS Files   -->
    {#<script src="/loginassets/js/core/jquery.3.2.1.min.js" type="text/javascript"></script>#}
    {#<script src="/loginassets/js/core/popper.min.js" type="text/javascript"></script>#}
    <script src="/loginassets/js/core/bootstrap.min.js" type="text/javascript"></script>
    <!--  Plugin for Switches, full documentation here:
http://www.jque.re/plugins/version3/bootstrap.switch/ -->
    <script src="/loginassets/js/plugins/bootstrap-switch.js"></script>
    <!--  Plugin for the Sliders, full documentation here: http://refreshless.com/nouislider/ -
->
    <script src="/loginassets/js/plugins/nouislider.min.js" type="text/javascript"></script>
    <!--  Plugin for the DatePicker, full documentation here:
https://github.com/uxsolutions/bootstrap-datepicker -->
    <script src="/loginassets/js/plugins/bootstrap-datepicker.js"
type="text/javascript"></script>
    <!-- Control Center for Now Ui Kit: parallax effects, scripts for the example pages etc -->
    <script src="/loginassets/js/now-ui-kit.js?v=1.1.0" type="text/javascript"></script>
    <!--  Notifications Plugin    -->
    <script src="/assets/js/bootstrap-notify.js"></script>

{% endblock %}
```

Dashboard.html.twig

```
{% extends 'base.html.twig' %}

{% block stylesheets %}

    {{ parent() }}

    <style>
        thead > tr, tbody{
            display:block;
        }

        td {
            width: 100%;
        }
    </style>

{% endblock %}

{% block content %}
    {#{{ dump()}}#}
```

```
    <div class="content">
        <div class="container-fluid">
            <div class="row">
                <div class="col-lg-3 col-sm-6">
                    <div class="card">
                        <div class="content">
                            <div class="row">
                                <div class="col-xs-5">
                                    <div class="icon-big icon-warning text-center">
                                        <i class="ti-user"></i>
                                    </div>
                                </div>
                                <div class="col-xs-7">
                                    <div class="numbers">
                                        <p>Employees</p>
                                        {{ stats.employee_nr }}
                                    </div>
                                </div>
                            </div>
                            <div class="footer">
                                <hr />
                                <div class="stats">
                                    <i class="ti-reload"></i> Total Employees
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
                <div class="col-lg-3 col-sm-6">
                    <div class="card">
                        <div class="content">
                            <div class="row">
                                <div class="col-xs-5">
                                    <div class="icon-big icon-info text-center">
                                        <i class="ti-target"></i>
                                    </div>
                                </div>
                                <div class="col-xs-7">
                                    <div class="numbers">
                                        <p>Active Jobs</p>
                                        {{ stats.open_jobs }}
                                    </div>
                                </div>
                            </div>
                            <div class="footer">
                                <hr />
                                <div class="stats">
                                    <i class="ti-calendar"></i> Updated now
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
                <div class="col-lg-3 col-sm-6">
                    <div class="card">
                        <div class="content">
                            <div class="row">
                                <div class="col-xs-5">
                                    <div class="icon-big icon-danger text-center">
                                        <i class="ti-timer"></i>
                                    </div>
                                </div>
                                <div class="col-xs-7">
                                    <div class="numbers">
                                        <p>Due Soon</p>
                                        {{ stats.due_soon }}
                                    </div>
```

```
                        </div>
                    </div>
                    <div class="footer">
                        <hr />
                        <div class="stats">
                            <i class="ti-timer"></i> Currently Active
                        </div>
                    </div>
                </div>
            </div>
            <div class="col-lg-3 col-sm-6">
                <div class="card">
                    <div class="content">
                        <div class="row">
                            <div class="col-xs-5">
                                <div class="icon-big icon-success text-center">
                                    <i class="ti-check"></i>
                                </div>
                            </div>
                            <div class="col-xs-7">
                                <div class="numbers">
                                    <p>Completed</p>
                                    {{ stats.recently }}
                                </div>
                            </div>
                        </div>
                        <div class="footer">
                            <hr />
                            <div class="stats">
                                <i class="ti-reload"></i> Updated now
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>


        <div class="row">
            <div class="col-md-12">
                <div class="card ">

                    <div class="header">
                        <h4 class="title">Last Week's Jobs</h4>
                        <p class="category">All jobs</p>
                    </div>

                    <div class="content">

                        <div id="chartLine1" class="ct-chart"></div>

                        <div class="footer">
                            <div class="chart-legend">
                                <i class="fa fa-circle text-info"></i> Opened Jobs
                                <i class="fa fa-circle text-warning"></i> Closed Jobs
                            </div>
                            <hr>
                            <div class="stats">
                                <i class="ti-check"></i> Data information certified
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
```

```
            <div class="row">
                <div class="col-md-12">
                    <div class="card ">

                        <div class="header">
                            <h4 class="title">Last Week's Top 5 Employees</h4>
                            <p class="category">Job Statistics</p>
                        </div>

                        <div class="content">

                            <div id="chartLine2" class="ct-chart"></div>
                            <div class="footer">
                                <div class="chart-legend">
                                    <i class="fa fa-circle text-info"></i> Completed Jobs
                                    <i class="fa fa-circle text-warning"></i> Assigned Jobs
                                </div>
                                <hr>
                                <div class="stats">
                                    <i class="ti-check"></i> Data information certified
                                </div>
                            </div>

                        </div>
                    </div>
                </div>
            </div>

            <div class="row">
                <div class="col-md-12">
                    <div class="card">
                        <div class="header">
                            <h4 class="title">Live feed</h4>
                            <p class="category">Latest jobs</p>
                        </div>
                        <div class="content">
                            {#<div class="table-full-width table-tasks" style="max-height:
200px; overflow: scroll; overflow-x: hidden; table-layout: fixed; border-collapse:
collapse;">#}
                            <div class="table-full-width table-tasks">
                                <table class="table">
                                    <thead style="width: 100%">
                                        <tr>
                                            <th style="float:left">Job</th>
                                            <th style="float:right">Created at</th>
                                        </tr>
                                    </thead>

                                    <tbody style="overflow: auto; height: 200px; display:
block" id="notification-table">

                                        {#<tr><td></td></tr>#}
                                    </tbody>
                                </table>
                            </div>
                            <div class="footer">
                                <hr>
                                <div class="stats">
                                    <i class="ti-check"></i> Data information certified
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
```

```
            <div class="row">
                <div class="col-md-6">
                    <div class="card">
                        <div class="header">
                            <h4 class="title">Job Statistics</h4>
                            <p class="category">Last Week's Jobs</p>
                        </div>
                        <div class="content">
                            <div id="chartLine3" class="ct-chart ct-perfect-fourth"></div>

                            <div class="footer">
                                <div class="chart-legend">
                                    <i class="fa fa-circle text-info"></i> Finished Before
                                    <i class="fa fa-circle text-danger"></i> Not Finished
                                    <i class="fa fa-circle text-warning"></i> Finished After
                                </div>
                                <hr>
                                <div class="stats">
                                    <i class="ti-timer"></i> Timeline regarding deadline
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
                <div class="col-md-6">
                    <div class="card">
                        <div class="header">
                            <h4 class="title">Users Behavior</h4>
                            <p class="category">24 Hours performance</p>
                        </div>
                        <div class="content">
                            <div id="chartLine4" class="ct-chart"></div>
                            <div class="footer">
                                <div class="chart-legend">


                                    {% for role_color in stats.top_roles_colors %}
                                        <i class="fa fa-circle" style="color:{{
role_color.color|raw }}"></i> {{ role_color.role|raw }}
                                    {#{% if(loop.index % 3 == 0) %}#}
                                        {#{{ '<br>'|raw }}#}
                                    {#{% endif %}#}
                                    {% endfor %}
                                </div>
                                <hr>
                                <div class="stats">
                                    <i class="ti-reload"></i> Updated 3 minutes ago
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>

{% endblock %}

{% block javascripts %}

    {{ parent() }}

    <script>

        $(document).ready(function () {
            $("#dashboard-active").parent().parent().addClass("active");
```

```
            myChart();

            function myChart(){


                // The first Chart

                var chart = new Chartist.Line('#chartLine1', {
                    labels: ['6 days ago', '5 days ago', '4 days ago', '3 days ago', '2 days
ago', '1 day ago', 'Today'],
                    series: [
                        [{% for open_jobs in stats.last_week_opened_jobs %}{{ open_jobs|raw
}},{% endfor %}],
                        [{% for closed_jobs in stats.last_week_closed_jobs %}{{ closed_jobs|raw
}},{% endfor %}],

                    ]
                }, {
                    low: 0
                });

                // Let's put a sequence number aside so we can use it in the event callbacks
                var seq = 0,
                    delays = 80,
                    durations = 500;

                // Once the chart is fully created we reset the sequence
                chart.on('created', function() {
                    seq = 0;
                });

                // On each drawn element by Chartist we use the Chartist.Svg API to trigger
SMIL animations
                chart.on('draw', function(data) {
                    seq++;

                    if(data.type === 'line') {
                        // If the drawn element is a line we do a simple opacity fade in. This
could also be achieved using CSS3 animations.
                        data.element.animate({
                            opacity: {
                                // The delay when we like to start the animation
                                begin: seq * delays + 1000,
                                // Duration of the animation
                                dur: durations,
                                // The value where the animation should start
                                from: 0,
                                // The value where it should end
                                to: 1
                            }
                        });
                    } else if(data.type === 'label' && data.axis === 'x') {
                        data.element.animate({
                            y: {
                                begin: seq * delays,
                                dur: durations,
                                from: data.y + 100,
                                to: data.y,
                                // We can specify an easing function from Chartist.Svg.Easing
                                easing: 'easeOutQuart'
                            }
                        });
                    } else if(data.type === 'label' && data.axis === 'y') {
                        data.element.animate({
                            x: {
                                begin: seq * delays,
                                dur: durations,
```

```
                                from: data.x - 100,
                                to: data.x,
                                easing: 'easeOutQuart'
                            }
                        });
                    } else if(data.type === 'point') {
                        data.element.animate({
                            x1: {
                                begin: seq * delays,
                                dur: durations,
                                from: data.x - 10,
                                to: data.x,
                                easing: 'easeOutQuart'
                            },
                            x2: {
                                begin: seq * delays,
                                dur: durations,
                                from: data.x - 10,
                                to: data.x,
                                easing: 'easeOutQuart'
                            },
                            opacity: {
                                begin: seq * delays,
                                dur: durations,
                                from: 0,
                                to: 1,
                                easing: 'easeOutQuart'
                            }
                        });
                    } else if(data.type === 'grid') {
                        // Using data.axis we get x or y which we can use to construct our
animation definition objects
                        var pos1Animation = {
                            begin: seq * delays,
                            dur: durations,
                            from: data[data.axis.units.pos + '1'] - 30,
                            to: data[data.axis.units.pos + '1'],
                            easing: 'easeOutQuart'
                        };

                        var pos2Animation = {
                            begin: seq * delays,
                            dur: durations,
                            from: data[data.axis.units.pos + '2'] - 100,
                            to: data[data.axis.units.pos + '2'],
                            easing: 'easeOutQuart'
                        };

                        var animations = {};
                        animations[data.axis.units.pos + '1'] = pos1Animation;
                        animations[data.axis.units.pos + '2'] = pos2Animation;
                        animations['opacity'] = {
                            begin: seq * delays,
                            dur: durations,
                            from: 0,
                            to: 1,
                            easing: 'easeOutQuart'
                        };

                        data.element.animate(animations);
                    }
                });

                // For the sake of the example we update the chart every time it's created with
a delay of 10 seconds
                chart.on('created', function() {
                    if(window.__exampleAnimateTimeout) {
```

```
                    clearTimeout(window.__exampleAnimateTimeout);
                    window.__exampleAnimateTimeout = null;
                }
                window.__exampleAnimateTimeout = setTimeout(chart.update.bind(chart),
12000);
            });

            // End of the first Chart




            // The second Chart

            var chart = new Chartist.Bar('#chartLine2', {
                labels: [{% for employee in stats.top_employees %}'{{ employee.name|raw }}'
+ ' {{ employee.surname|raw }}',{% endfor %}],
                series: [
                    [{% for employee in stats.top_employees %}{{ employee.closed_jobs|raw
}},{% endfor %}],
                    [{% for employee in stats.top_employees %}{{ employee.assigned_jobs|raw
}},{% endfor %}],
                ]
            }, {
                // stackBars: true,
                axisY: {
                    labelInterpolationFnc: function(value) {
                        return value;
                    }
                }
            }).on('draw', function(data) {
                if(data.type === 'bar') {
                    data.element.attr({
                        style: 'stroke-width: 50px'
                    });
                }
            });

            chart.on('draw', function (data) {
                if (data.type === 'bar') {
                    data.element.attr({
                        style: 'stroke-width: 0px'
                    });
                    var strokeWidth = 10;

                    if (data.seriesIndex === 0) {
                        data.element.animate({
                            y2: {
                                begin: 0,
                                dur: 500,
                                from: data.y1,
                                to: data.y2,
                                easing: Chartist.Svg.Easing.easeOutSine,
                            },
                            'stroke-width': {
                                begin: 0,
                                dur: 1,
                                from: 1,
                                to: strokeWidth,
                                fill: 'freeze',
                            }
                        }, false);
                    }

                    if (data.seriesIndex === 1) {
                        data.element.animate({
                            y2: {
```

```
                                    begin: 500,
                                    dur: 500,
                                    from: data.y1,
                                    to: data.y2,
                                    easing: Chartist.Svg.Easing.easeOutSine,
                                },
                                'stroke-width': {
                                    begin: 500,
                                    dur: 1,
                                    from: 0,
                                    to: strokeWidth,
                                    fill: 'freeze',
                                }
                            }, false);
                        }
                    }
                });
                // End of the Second Chart


                // Third Chart (PIE CHART)


                var sum = {{ stats.finishedBeforeDeadline }} + {{ stats.finishedAfterDeadline
}} + {{ stats.notFinished }};
                var FBD = {{ stats.finishedBeforeDeadline }} * (1/sum) * 100;
                var FAD = {{ stats.finishedAfterDeadline }} * (1/sum) * 100;
                var NF = {{ stats.notFinished }} * (1/sum) * 100;

                FBD = FBD.toFixed(2);
                FAD = FAD.toFixed(2);
                NF = NF.toFixed(2);

                var chart = new Chartist.Pie('#chartLine3', {
                    labels: [FBD + '%', FAD + '%', NF + '%'],
                    series: [{{ stats.finishedBeforeDeadline }}, {{ stats.finishedAfterDeadline
}}, {{ stats.notFinished }}],
                }, {
                    showLabel: true,
                    donut: true
                });

                chart.on('draw', function(data) {
                    if(data.type === 'slice') {
                        // Get the total path length in order to use for dash array animation
                        var pathLength = data.element._node.getTotalLength();

                        // Set a dasharray that matches the path length as prerequisite to
animate dashoffset
                        data.element.attr({
                            'stroke-dasharray': pathLength + 'px ' + pathLength + 'px'
                        });

                        // Create animation definition while also assigning an ID to the
animation for later sync usage
                        var animationDefinition = {
                            'stroke-dashoffset': {
                                id: 'anim' + data.index,
                                dur: 1000,
                                from: -pathLength + 'px',
                                to:  '0px',
                                easing: Chartist.Svg.Easing.easeOutQuint,
                                // We need to use `fill: 'freeze'` otherwise our animation will
fall back to initial (not visible)
                                fill: 'freeze'
                            }
                        };
```

```
                        // If this was not the first slice, we need to time the animation so
that it uses the end sync event of the previous animation
                        if(data.index !== 0) {
                            animationDefinition['stroke-dashoffset'].begin = 'anim' +
(data.index - 1) + '.end';
                        }

                        // We need to set an initial value before the animation starts as we
are not in guided mode which would do that for us
                        data.element.attr({
                            'stroke-dashoffset': -pathLength + 'px'
                        });

                        // We can't use guided mode as the animations need to rely on setting
begin manually
                        // See http://gionkunz.github.io/chartist-js/api-
documentation.html#chartistsvg-function-animate
                        data.element.animate(animationDefinition, false);
                    }
                });

                // For the sake of the example we update the chart every time it's created with
a delay of 8 seconds
                chart.on('created', function() {
                    if(window.__anim21278907124) {
                        clearTimeout(window.__anim21278907124);
                        window.__anim21278907124 = null;
                    }
                    window.__anim21278907124 = setTimeout(chart.update.bind(chart), 10000);
                });
            }

            // Last Chart (USER BEHAVIOUR)
            var chart = new Chartist.Pie('#chartLine4', {
                series: [{% for role in stats.top_roles %}{{ role.role_count|raw }},{% endfor
%}],
                labels: [{% for role in stats.top_roles %}'{{ role.role_count|raw }}',{% endfor
%}]

            }, {
                donut: true,
                showLabel: true
            });

            chart.on('draw', function(data) {
                if(data.type === 'slice') {
                    // Get the total path length in order to use for dash array animation
                    var pathLength = data.element._node.getTotalLength();

                    // Set a dasharray that matches the path length as prerequisite to animate
dashoffset
                    data.element.attr({
                        'stroke-dasharray': pathLength + 'px ' + pathLength + 'px'
                    });

                    // Create animation definition while also assigning an ID to the animation
for later sync usage
                    var animationDefinition = {
                        'stroke-dashoffset': {
                            id: 'anim' + data.index,
                            dur: 1000,
                            from: -pathLength + 'px',
                            to:  '0px',
                            easing: Chartist.Svg.Easing.easeOutQuint,
                            // We need to use `fill: 'freeze'` otherwise our animation will
fall back to initial (not visible)
```

```
                            fill: 'freeze'
                        }
                    };

                    // If this was not the first slice, we need to time the animation so that
it uses the end sync event of the previous animation
                    if(data.index !== 0) {
                        animationDefinition['stroke-dashoffset'].begin = 'anim' + (data.index -
1) + '.end';
                    }

                    // We need to set an initial value before the animation starts as we are
not in guided mode which would do that for us
                    data.element.attr({
                        'stroke-dashoffset': -pathLength + 'px'
                    });

                    // We can't use guided mode as the animations need to rely on setting begin
manually
                    // See http://gionkunz.github.io/chartist-js/api-
documentation.html#chartistsvg-function-animate
                    data.element.animate(animationDefinition, false);
                }
            });

// For the sake of the example we update the chart every time it's created with a delay of 8
seconds
            chart.on('created', function() {
                if(window.__anim21278907124) {
                    clearTimeout(window.__anim21278907124);
                    window.__anim21278907124 = null;
                }
                window.__anim21278907124 = setTimeout(chart.update.bind(chart), 10000);
            });
            // End


            // Notifications

            setInterval(function(){
                $.ajax({
                    type: "POST",
                    url: '/get/notification',

                    success: function (msg) {
                        var notifications = JSON.parse(msg);

                        $('#notification-table').html('');

                        notifications.forEach(function(element){
                            $('#notification-table').append(
                                '<tr style="width: 100%">' +
                                '<td style="width: 83%">' + element.content +  '</td>' + '<td>'
+ element.created_at + '</td>' +
                                '</tr>'
                            );
                        });
                    }
                });
            }, 5000);

        });
    </script>

{% endblock %}
```

Base.html.twig

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8"/>
    <title>{% block title %}Welcome!{% endblock %}</title>
    {% block stylesheets %}
        <meta charset="utf-8" />
        <link rel="apple-touch-icon" sizes="76x76" href="/paper-assets/img/apple-icon.png">
        <link rel="icon" type="image/png" sizes="96x96" href="/paper-assets/img/favicon.png">
        <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />

        <title>Paper Dashboard by Creative Tim</title>

        <meta content='width=device-width, initial-scale=1.0, maximum-scale=1.0, user-
scalable=0' name='viewport' />
        <meta name="viewport" content="width=device-width" />


        <!-- Bootstrap core CSS     -->
        <link href="/paper-assets/css/bootstrap.min.css" rel="stylesheet" />

        <!-- Animation library for notifications   -->
        <link href="/paper-assets/css/animate.min.css" rel="stylesheet"/>

        <!--  Paper Dashboard core CSS    -->
        <link href="/paper-assets/css/paper-dashboard.css" rel="stylesheet"/>

        <!-- css for the employee pagination -->
        <link rel="stylesheet" type="text/css" href="/paper-assets/css/pagination.css">

        <!-- css for search bar -->
        <link rel="stylesheet" type="text/css" href="/paper-assets/css/search.css">

        <!--  CSS for Demo Purpose, don't include it in your project     -->
        <link href="/paper-assets/css/demo.css" rel="stylesheet" />


        <!--  Fonts and icons     -->
        <link href="http://maxcdn.bootstrapcdn.com/font-awesome/latest/css/font-
awesome.min.css" rel="stylesheet">
        <link href='https://fonts.googleapis.com/css?family=Muli:400,300' rel='stylesheet'
type='text/css'>
        <link href="/paper-assets/css/themify-icons.css" rel="stylesheet">
    {% endblock %}


</head>
<body>
{% block body %}
    <div class="wrapper">

        {% block sidebar %}
            <div class="sidebar" data-background-color="black" data-active-color="danger">

                <!--
                    Tip 1: you can change the color of the sidebar's background using: data-
background-color="white | black"
                    Tip 2: you can change the color of the active button using the data-active-
color="primary | info | success | warning | danger"
                -->

                <div class="sidebar-wrapper">
                    <div class="logo">
```

```
                    <a href="" class="simple-text">
                        Meet<span style="color: #eb5e28;font-size: 12px">LY</span>
                    </a>
                </div>

                <ul class="nav">
                    <li>
                        <a href="{{ path('dashboard') }}">
                            <i class="ti-panel"></i>
                            <p id="dashboard-active">Dashboard</p>
                        </a>
                    </li>

                    <li>
                        <a href="{{ path('maps') }}">
                            <i class="ti-map"></i>
                            <p id="map-active">Map</p>
                        </a>
                    </li>

                    <li>
                        <a href="{{ path('jobs') }}">
                            <i class="ti-book"></i>
                            <p id="jobs-active">Jobs</p>
                        </a>
                    </li>

                    <li>
                        <a href="{{ path('employee') }}">
                            <i class="ti-user"></i>
                            <p id="employees-active">Employees</p>
                        </a>
                    </li>

                    <li>
                        <a href="{{ path('reports') }}">
                            <i class="ti-view-list-alt"></i>
                            <p id="reports-active">Reports</p>
                        </a>
                    </li>

                    <li>
                        <a href="{{ path('user') }}">
                            <i class="ti-settings"></i>
                            <p id="settings-active">Settings</p>
                        </a>
                    </li>

                </ul>
            </div>
        </div>
    {% endblock %}

    <div class="main-panel">
        {% block navbar %}
            <nav class="navbar navbar-default">
                <div class="container-fluid">
                    <div class="navbar-header">
                        <button type="button" class="navbar-toggle">
                            <span class="sr-only">Toggle navigation</span>
                            <span class="icon-bar bar1"></span>
                            <span class="icon-bar bar2"></span>
                            <span class="icon-bar bar3"></span>
                        </button>
                        <a class="navbar-brand" href="#">
                            <i class="ti-panel"></i>
                            Dashboard
```

```
                        </a>
                    </div>
                    <div class="collapse navbar-collapse">
                        <ul class="nav navbar-nav navbar-right">
                            <li>
                                <a href="{{ path('logout') }}">
                                    <i class="ti-key"></i>
                                    <p>Log out</p>
                                </a>
                            </li>
                        </ul>

                    </div>
                </div>
            </nav>
        {% endblock %}

        {% block content %}{% endblock %}

        {% block footer %}
            <footer class="footer">
                <div class="container-fluid">
                    <nav class="pull-left">
                        <ul>

                            <li>
                                <a href="#">
                                    KuDa
                                </a>
                            </li>
                            <li>
                                <a href="#">
                                    Blog
                                </a>
                            </li>
                            <li>
                                <a href="#">
                                    Licenses
                                </a>
                            </li>
                        </ul>
                    </nav>
                    <div class="copyright pull-right">
                        &copy; <script>document.write(new Date().getFullYear())</script>,
made with <i class="fa fa-heart heart"></i> by <a href="#">KuDa</a>
                    </div>
                </div>
            </footer>
        {% endblock %}

    </div>

</div>

{% block modal %}
{% endblock %}

{% endblock %}




{% block javascripts %}

    <script src="/paper-assets/js/jquery-1.10.2.js" type="text/javascript"></script>
    <script src="/paper-assets/js/bootstrap.min.js" type="text/javascript"></script>
```

```
    <!--  Checkbox, Radio & Switch Plugins -->
    <script src="/paper-assets/js/bootstrap-checkbox-radio.js"></script>

    <!--  Charts Plugin -->
    <script src="/paper-assets/js/chartist.min.js"></script>

    <!--  Notifications Plugin    -->
    <script src="/paper-assets/js/bootstrap-notify.js"></script>

    <!-- Paper Dashboard Core javascript and methods for Demo purpose -->
    <script src="/paper-assets/js/paper-dashboard.js"></script>

    <!-- Paper Dashboard DEMO methods, don't include it in your project! -->
    <script src="/paper-assets/js/demo.js"></script>

    <!-- Scroll Bar Javascript -->
    <script src="/paper-assets/js/scrollbar.js"></script>



    <!--  Google Maps Plugin    -->
    <script type="text/javascript"

src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCq8LwaJFFVfHBwqX7PGa5YolKums8Xa_E"></scr
ipt>
    <script
src="https://developers.google.com/maps/documentation/javascript/examples/markerclusterer/marke
rclusterer.js"></script>

    <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

    <script type="text/javascript">
        $(document).ready(function () {

            $('.sidebar-wrapper').on('click', 'li', function () {
                $(".active").removeClass('active');
                $(this).addClass('active');
                console.log($(".active"));
                console.log($(this));
            });

            {% if user.name is defined %}
            $.notify({
                icon: 'ti-user',
                message: "Welcome {{ user.name }} {{ user.surname }}!"

            }, {
                type: 'danger',
                timer: 4000
            });
            {% endif %}

        });
    </script>

{% endblock %}
</body>
</html>
```

User.html.twig

```
{% extends 'base.html.twig' %}
```

```
{% block content %}

        <div class="content">
            <div class="container-fluid">
                <div class="row">
                    <div class="col-md-1"></div>

                    <div class="col-md-10">
                        <div class="card">
                            <div class="header">
                                <h4 class="title">Edit Profile</h4>
                            </div>
                            <div class="content">
                                <form action="{{ path('update') }}" method="POST">
                                    <div class="row">
                                        <div class="col-md-4">
                                            <div class="form-group">
                                                <label>Company (disabled)</label>
                                                <input type="text" class="form-control"
disabled placeholder="Company" value="Software Engineering Inc.">
                                            </div>
                                        </div>
                                        <div class="col-md-4">
                                            <div class="form-group">
                                                <label>Username</label>
                                                <input type="text" name="username" class="form-
control" placeholder="Username" value="{{ user.username }}" style="pointer-
events:none;background:#F5F5F5;cursor: not-allowed;">
                                            </div>
                                        </div>
                                        <div class="col-md-4">
                                            <div class="form-group">
                                                <label for="exampleInputEmail1">Email
address</label>
                                                <input type="email" name="email" class="form-
control" placeholder="Email" value="{{ user.email }}" required>
                                            </div>
                                        </div>
                                    </div>

                                    <div class="row">
                                        <div class="col-md-6">
                                            <div class="form-group">
                                                <label>First Name</label>
                                                <input type="text" name="name" class="form-
control" placeholder="Company" value="{{ user.name }}" required>
                                            </div>
                                        </div>
                                        <div class="col-md-6">
                                            <div class="form-group">
                                                <label>Last Name</label>
                                                <input type="text" name="surname" class="form-
control" placeholder="Last Name" value="{{ user.surname }}" required>
                                            </div>
                                        </div>
                                    </div>

                                    <div class="row">
                                        <div class="col-md-12">
                                            <div class="form-group">
                                                <label>Phone Number</label>
                                                <input type="text" name="phone_nr" class="form-
control" placeholder="Birthday" value="{{ user.getPhoneNr }}" required>
                                            </div>
                                        </div>
                                    </div>
```

```
                                    <div class="row">
                                        <div class="col-md-4">
                                            <div class="form-group">
                                                <label>Birthday</label>
                                                <input type="datetime" name="birthday"
class="form-control" placeholder="Birthday" value="{{ user.birthday|date('Y-m-d') }}" required>
                                            </div>
                                        </div>
                                        <div class="col-md-4">
                                            <div class="form-group">
                                                <label>Role</label>
                                                <input type="text" class="form-control"
name="role" placeholder="Role" value="{{ user.role }}" required>
                                            </div>
                                        </div>
                                        <div class="col-md-4">
                                            <div class="form-group">
                                                <label>Salary</label>
                                                <input type="text" class="form-control"
name="salary" placeholder="Salary" value="{{ user.salary }}" style="pointer-
events:none;background:#F5F5F5;cursor:  not-allowed">
                                            </div>
                                        </div>
                                    </div>

                                    <div class="row">
                                        <div class="col-md-6">
                                            <div class="form-group">
                                                <label>New Password</label>
                                                <input type="password" name="password"
class="form-control" placeholder="**********" >
                                            </div>
                                        </div>
                                        <div class="col-md-6">
                                            <div class="form-group">
                                                <label>Confirm Password</label>
                                                <input type="password" name="new_password"
class="form-control" placeholder="*********" >
                                            </div>
                                        </div>
                                    </div>

                                    <button type="submit" class="btn btn-outline-danger pull-
right">Update Profile</button>
                                    <div class="clearfix"></div>
                                </form>
                            </div>
                        </div>
                    </div>
                    <div class="col-md-1"></div>

                </div>
            </div>
        </div>
{% endblock %}

{% block javascripts %}

    {{ parent() }}

    <script>

        $(document).ready(function(){
            $("#settings-active").parent().parent().addClass("active");
        });

    </script>
```

```
{% endblock %}
```

Jobs.html.twig

```
{% extends 'base.html.twig' %}

{% block title %}Jobs{% endblock %}

{% block stylesheets %}
    {{ parent() }}

    <link href="{{ asset('assets/css/job/job.css') }}" rel="stylesheet"/>
    <link
href="https://fonts.googleapis.com/css?family=Bentham|Playfair+Display|Raleway:100,200,400,500|
Suranna|Trocchi" rel="stylesheet">

    <style>
        /*#easyPaginate {width:300px;}*/
        .easyPaginateNav a {padding:1px;}
        .easyPaginateNav a.current {font-weight:bold;text-decoration:underline;}
        .nav-pills>li.active>a, .nav-pills>li.active>a:focus, .nav-pills>li.active>a:hover {
            color: #fff !important;
            background-color: #d9534f !important;
        }

    </style>
{% endblock %}


{% block navbar %}

    <nav class="navbar navbar-default">
        <div class="container-fluid">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle">
                    <span class="sr-only">Toggle navigation</span>
                    <span class="icon-bar bar1"></span>
                    <span class="icon-bar bar2"></span>
                    <span class="icon-bar bar3"></span>
                </button>
                <a class="navbar-brand" href="#">
                    <i class="ti-book"></i>
                    Jobs
                </a>
            </div>
            <div class="collapse navbar-collapse">
                <ul class="nav navbar-nav navbar-left">

                    <li>
                        <a class="search-btn">
                            {#<div class="form-group">#}
                                <input type="text" name="searchJob" class="form-control"
id="my-search" placeholder="Search Job">
                            {#</div>#}
                        </a>
                    </li>

                    <li>
                        <a>
                            <i class="fa fa-search"></i>
                        </a>
                    </li>
```

```
                </ul>
                <ul class="nav navbar-nav navbar-right">
                    <li>
                        <a href="{{ path('logout') }}">
                            <i class="ti-key"></i>
                            <p>Log out</p>
                        </a>
                    </li>
                </ul>
            </div>
        </div>
    </nav>
{% endblock %}



{% block content %}
    <div class="content">
        <div class="container-fluid">
            <div class="row">
                <div class="col-md-12">

                    <div id="easyPaginate">

                        {% for job in jobs %}

                            <page>
                            <div class="card">
                                <div class="card-header">
                                    <ul class="nav nav-pills nav-justified">
                                        {#nav nav-tabs text-centered#}
                                        <li class="col-md-3 nav-item">
                                            <a class="nav-link active-tab" data-toggle="tab"
href="#job{{ job.job_id }}">Job{{ job.job_id }}</a>
                                        </li>
                                        <li class="col-md-3 nav-item">
                                            <a class="nav-link"  data-toggle="tab"
href="#answer{{ job.job_id }}">Answer</a>
                                        </li>
                                    </ul>

                                </div>
                                <div class="content card-body">

                                    <div class="tab-content" style="height: 380px">
                                        <div id="job{{ job.job_id }}" class="tab-pane fade in">
                                            <div class="job-wrapper">
                                                <div class="job-img">
                                                    <img class="img-job" height="350"
width="327" src="{{ asset('assets/img/job_images/job-photo.png') }}">
                                                    <div class="img-txt-center">
                                                        {{ job.description }}
                                                    </div>
                                                </div>
                                                <div class="job-info" id="style2"
style="overflow: scroll;overflow-x: hidden">
                                                    <div class="job-text">
                                                        <h1 class="job-h1 text-center">{{
job.title }}</h1>

                                                        <h2 class="job-h2 text-center">by {{
job.name }} {{ job.surname }}</h2>

                                                        <p class="job-p text-center"
style="border-bottom: 1px solid #cccccc"><strong>Priority:</strong>
                                                            {% if job.priority == 0 %}
                                                                Low
                                                            {% elseif job.priority == 1 %}
                                                                Medium
```

```
                                                    {% else %}
                                                        High
                                                    {% endif %}
                                                </p>
                                                <p class="job-p text-center"
style="border-bottom: 1px solid #cccccc"><strong>Difficulty:</strong>
                                                    {% if job.job_diff == 0 %}
                                                        Easy
                                                    {% elseif job.job_diff == 1 %}
                                                        Challenging
                                                    {% elseif job.job_diff == 2 %}
                                                        Hard
                                                    {% else %}
                                                        Extreme
                                                    {% endif %}
                                                </p>
                                                <p class="job-p text-center"
style="border-bottom: 1px solid #cccccc"><strong>Reward:</strong> ${{ job.reward }}</p>
                                                <p class="job-p text-center"
style="border-bottom: 1px solid #cccccc"><strong>Deadline:</strong> {{ job.deadline }}</p>
                                            </div>
                                            <div class="job-price-btn" style="display:
flex; justify-content: center; align-items: center;margin-top:5px">
                                                <button class="job-btn btn-danger
delete-job-btn" onclick="appendToModal({{ job.job_id }}, {{ job.answer_id }})" type="button"
data-toggle="modal" id="delete-{{ job.job_id }}" data-target="#delete-modal"
style="margin:0">Delete</button>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                                <div id="answer{{ job.job_id }}" class="tab-pane fade"
style="display: flex; justify-content: center; align-items: center;">
                                    {% if job.response is not null %}
                                        <div class="job-wrapper">
                                            <div class="job-img">
                                                <img class="img-job" height="350"
width="327" src="{{ asset('assets/img/job_images/answer-photo.jpg') }}">
                                                <div class="img-txt-center">
                                                    {{ job.description }}
                                                </div>
                                            </div>
                                            <div class="job-info" id="style2"
style="overflow: scroll; overflow-x: hidden">
                                                <div class="job-text">
                                                    <h1 class="job-h1 text-center">{{
job.title }}</h1>
                                                    <h2 class="job-h2 text-center">by
{{ job.name }} {{ job.surname }}</h2>
                                                    {#<h1 class='job-h1 text-center'
id='response'>Response</h1><br>#}
                                                    <p class='job-p' id="response-
questions"></p>
                                                        <script>
                                                            var show_response = "";

//                                                            var map = {"Question 1" :
50, "Question2" : "yes", "Question23" : "yes", "Questio1n23" : "yes", "Questaion23" : "yes",
"Questdion23" : "yes", "Ques123tion23" : "yes", "Question22" : "yes", "Questi1on2" : "yes"};



                                            var response = '{{ response }}';
                                            var map =
JSON.parse(response.replace(/&quot;/g, '"'));

                                            console.log(map);
```

```
                                                                map.forEach(function
(element) {

                                                                    show_response += "<p
style='border-bottom:1px solid #cccccc;margin:0' class='job-b text-center'><strong>" +
element[0] + "? \t </strong>" + element[1] + "</p><br>";

                                                                });

                                                                // $('#response').html('');

document.getElementById('response-questions').innerHTML = document.getElementById('response-
questions').innerHTML + show_response;
                                                    //append(show_response);
                                                    console.log(show_response);
                                                            </script>
                                                        <p class='job-p text-center'
style='margin-bottom: 15px;border-bottom:1px solid #cccccc'><strong>Answered At:</strong>
                                                            {{ job.answered_at }}
                                                        </p>
                                                    </div>
                                                </div>
                                            </div>
                                        {% else %}
                                            <h3 style='position:absolute;top:40%'>Sorry!
This job has not been answered yet!</h3>
                                        {% endif %}
                                    </div>
                                </div>
                                <div class="footer">
                                    <hr>
                                    <div class="stats">
                                        <i class="ti-check"></i> Job Information
                                    </div>
                                </div>
                            </div>
                        </div>
                        </page>
                    {% endfor %}

                </div>
            </div>
        </div>
    </div>
</div>

{% endblock %}

{% block modal %}

    <div id="delete-modal" class="modal fade" tabindex="-1" role="dialog" aria-
labelledby="myModalLabel"
         aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header">
                    <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">×</button>
                    <center>
                        <h3 id="myModalLabel">Delete Job</h3>
                    </center>
                </div>

                <div class="modal-body" id="myWizard">
                    <div class="text-center" style="margin-bottom: 0px !important;">
                        <div class="content">
```

```
                            <div class="row">
                                <div class="col-md-12">
                                    <div class="form-group">
                                        <h3>Do you really want to delete this job?</h3>
                                        <h7><strong>Note:</strong> The answer for this job will
be deleted aswell.</h7>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                    <div class="modal-footer">
                        <div class="col-md-1"></div>
                        <button class="btn btn-danger col-md-5"
onclick="deleteJob()">Yes</button>
                        <button class="btn btn col-md-5 search-btn-modal" data-dismiss="modal"
data-toggle="tab">No</button>

                        <div class="col-md-1"></div>
                    </div>
                </div>
            </div>
        </div>
    </div>

{% endblock %}

{% block javascripts %}

    {{ parent() }}

    <script src="{{ asset('assets/js/jobs/easyPaginate.js') }}"></script>

    <script>

        var test = {{ jobs|json_encode|raw }};

        test.forEach(function (value) {
            // console.log(value);
        });



        $(document).ready(function () {

            $('.active').removeClass('active');
            $("#jobs-active").parent().parent().addClass('active');

            $('.active-tab').tab('show');

            // a.twbsPagination();

            $('#easyPaginate').easyPaginate({
                paginateElement: 'page',
                elementsPerPage: 5,
                effect: 'climb',
                firstButtonText: 'First',
                lastButtonText: 'Last',
                nextButtonText: 'Next',
                prevButtonText: 'Prev'
            });


            // Search Jobs

            $("#my-search").keyup(function() {
```

```
                var jobTitle = $(this).val();

                delay(function(){

                    // alert(jobTitle);

                    if (jobTitle !== '') {
                        $.ajax({
                            type: 'POST',
                            url: '/search/jobs',
                            data: {
                                'job-title' : jobTitle
                            },
                            success: function (result) {

                                console.log(JSON.parse(result));

                                var array = JSON.parse(result);

                                $('#easyPaginate').html('');

                                array.forEach(function (element) {

                                    var answer = "";
                                    var priority = "";
                                    var difficulty = "";

                                    if(element.priority === "0"){
                                        priority = "Low";
                                    } else if(element.priority === "1"){
                                        priority = "Medium";
                                    } else {
                                        priority = "High";
                                    }

                                    if(element.job_diff === "0"){
                                        difficulty = "Easy";
                                    } else if(element.job_diff === "1"){
                                        difficulty = "Challenging";
                                    } else if(element.job_diff === "2"){
                                        difficulty = "Hard";
                                    } else {
                                        difficulty = "Extreme";
                                    }

                                    var show_response = "";

//                                      var map = {"Question 1" : 50, "Question2" : "yes",
"Question23" : "yes", "Questio1n23" : "yes", "Questaion23" : "yes", "Questdion23" : "yes",
"Ques123tion23" : "yes", "Question22" : "yes", "Questi1on2" : "yes"};

                                    if(element.response !== null) {
                                var map = new Map(JSON.parse(element.response));

                                        Object.keys(map).forEach(function (key) {

                                            var value = map[key];
                                            show_response += "<p style='border-bottom:1px solid
#cccccc;margin:0' class='job-b text-center'><strong>" + key + "? \t </strong>" + value +
"</p><br>";

                                        });
                                    }

                                    if(element.answer_id != null){

                                        answer = "<div class='job-wrapper'>\n" +
```

```
"                                                    <div class='job-img'>\n" +
"                                                        <img class='img-job' height='350'
width='327' src='{{ asset('assets/img/job_images/answer-photo.jpg') }}'>\n" +
"                                                    <div class='img-txt-center'>\n" +
"                                                        " + element.description + "\n" +
"                                                    </div>\n" +
"                                                </div>\n" +
"                                                <div class='job-info' id='style2'
style='overflow:scroll; overflow-x:hidden;'>\n" +
"                                                    <div class='job-text'>\n" +
"                                                        <h1 class='job-h1 text-center'>" +
element.title + "</h1>\n" +
"                                                        <h2 class='job-h2 text-center'>by
" + element.name + " " + element.surname + "</h2>\n" +
"                                                        <p class='job-p text-center'>" +
"                                                            " + show_response + "\n" +
"                                                        </p>\n" +
"                                                        <p class='job-p text-center'
style='margin-bottom: 15px;border-bottom:1px solid #cccccc'><strong>Answered At:</strong>\n" +
"                                                            " + element.answered_at + "\n"
+
"                                                        </p>\n" +
"                                                    </div>\n" +
"                                                </div>\n" +
"                                            </div>";

                                    } else {
                                        answer = "<h3 style='position:absolute;top:40%'>Sorry!
This job has not been answered yet!</h3>";
                                    }



                                    $('#easyPaginate').append("<page>\n" +
                                "                                        <div class='card'>\n" +
                                "                                            <div class='card-header'>\n" +
                                "                                                <ul class='nav nav-pills
nav-justified'>\n" +
                                "                                                    <li class='col-md-3
nav-item'>\n" +
                                "                                                        <a class='nav-link
active-tab' data-toggle='tab' href='#job" + element.job_id + "'>Job" + element.job_id +
"</a>\n" +
                                "                                                    </li>\n" +
                                "                                                    <li class='col-md-3
nav-item'>\n" +
                                "                                                        <a class='nav-
link' data-toggle='tab' href='#answer" + element.job_id + "'>Answer</a>\n" +
                                "                                                    </li>\n" +
                                "                                                </ul>\n" +
                                "\n" +
                                "                                            </div>\n" +
                                "                                            <div class='content card-
body'>\n" +
                                "\n" +
                                "                                                <div class='tab-content'
style='height: 380px'>\n" +
                                "                                                    <div id='job" +
element.job_id + "' class='tab-pane fade in'>\n" +
                                "                                                        <div class='job-
wrapper'>\n" +
                                "                                                            <div
class='job-img'>\n" +
                                "                                                                <img
class='img-job' height='350' width='327' src='{{ asset('assets/img/job_images/job-photo.png')
}}'>\n" +
                                "                                                                <div
```

```
class='img-txt-center'>\n" +
                                "                                                                                " +
element.description + "\n" +
                                "                                                                            </div>\n"
+
                                "                                                                        </div>\n" +
                                "                                                                        <div
class='job-info' id='style2' style='overflow:scroll; overflow-x:hidden;'>\n" +
                                "                                                                            <div
class='job-text'>\n" +
                                "                                                                                <h1
class='job-h1 text-center'>" + element.title + "</h1>\n" +
                                "                                                                                <h2
class='job-h2 text-center'>by " + element.name + " " + element.surname + "</h2>\n" +
                                "                                                                                <p
class='job-p text-center' style='border-bottom: 1px solid
#cccccc'><strong>Priority:</strong>\n" + priority +
                                "
</p>\n" +
                                "                                                                                <p
class='job-p text-center' style='border-bottom: 1px solid
#cccccc'><strong>Difficulty:</strong>\n" + difficulty +
                                "
</p>\n" +
                                "                                                                                <p
class='job-p text-center' style='border-bottom: 1px solid #cccccc'><strong>Reward:</strong> $"
+ element.reward + "</p>\n" +
                                "                                                                                <p
class='job-p text-center' style='border-bottom: 1px solid #cccccc'><strong>Deadline:</strong> "
+ element.deadline + "</p>\n" +
                                "                                                                            </div>\n"
+
                                "                                                                            <div
class='job-price-btn' style='display: flex; justify-content: center; align-items:
center;margin-top:5px'>\n" +
                                "
<button class='job-btn btn-danger delete-job-btn' onclick='appendToModal(" + element.job_id +
"," + element.answer_id + ")' type='button' data-toggle='modal' id='delete-" + element.job_id +
"' data-target='#delete-modal' style='margin:0'>Delete</button>\n" +
                                "                                                                            </div>\n"
+
                                "                                                                        </div>\n" +
                                "                                                                    </div>\n" +
                                "                                                                </div>\n" +
                                "                                                                <div id='answer" +
element.job_id + "' class='tab-pane fade'  style='display: flex; justify-content: center;
align-items: center;'>\n" +
                                "                                                                    answer +
                                "                                                                </div>" +
                                "                                                            </div>\n" +
                                "                                                            <div class='footer'>\n" +
                                "                                                                <hr>\n" +
                                "                                                                <div class='stats'>\n"
+
                                "                                                                    <i class='ti-
check'></i> Job Information\n" +
                                "                                                                </div>\n" +
                                "                                                            </div>\n" +
                                "                                                        </div>\n" +
                                "                                                    </div>\n" +
                                "                                                    </page>");

                                    $('.active-tab').tab('show');

                                });
```

```
                                $('.easyPaginateNav').remove();

                                $('#easyPaginate').easyPaginate({
                                    paginateElement: 'page',
                                    elementsPerPage: 5,
                                    effect: 'climb',
                                    firstButtonText: 'First',
                                    lastButtonText: 'Last',
                                    nextButtonText: 'Next',
                                    prevButtonText: 'Prev'
                                });
                            }
                        });
                    }

                }, 1000);
            });
        });


        var delay = (function(){
            var timer = 0;
            return function(callback, ms){
                clearTimeout (timer);
                timer = setTimeout(callback, ms);
            };
        })();


        // Delete Job

        function appendToModal(jobId, answerId) {
            $('#delete-job-id').remove();
            $('#delete-answer-id').remove();
            $('#myWizard').append('<p id="delete-job-id" style="display: none">' + jobId +
'</p>');
            $('#myWizard').append('<p id="delete-answer-id" style="display: none">' + answerId
+ '</p>');
        }

        function deleteJob() {


            var jobId = $('#delete-job-id').html();
            var answerId = $('#delete-answer-id').html();
            // alert(jobId);
            // alert(answerId);

            $.ajax({
                type: 'POST',
                url: '/delete/jobs',
                data: {
                    'job-id': jobId,
                    'answer-id': answerId
                },
                success: function (result) {
                    location.reload();
                }
            });
        }


    </script>

{% endblock %}
```

Maps.html.twig

```
{% extends 'base.html.twig' %}

{% block title %}Map{% endblock %}
{% block stylesheets %}
    {{ parent() }}

    <link href="{{ asset('assets/css/dynamicForm.css') }}" rel="stylesheet"/>

{% endblock %}

    {% block navbar %}

        <nav class="navbar navbar-default">
            <div class="container-fluid">
                <div class="navbar-header">
                    <button type="button" class="navbar-toggle">
                        <span class="sr-only">Toggle navigation</span>
                        <span class="icon-bar bar1"></span>
                        <span class="icon-bar bar2"></span>
                        <span class="icon-bar bar3"></span>
                    </button>
                    <a class="navbar-brand" href="#">
                        <i class="ti-map"></i>
                        Map
                    </a>
                </div>
                <div class="collapse navbar-collapse">
                    <ul class="nav navbar-nav navbar-left">
                        <li>
                            <a href="#" class="" data-toggle="modal" data-target="#myModal">
                                <i class="fa fa-plus"></i>
                                <p class="hidden-lg hidden-md">Map</p>
                            </a>
                        </li>
                        {#<li>#}
                            {#<a class="search-btn">#}
                                {#<i class="fa fa-search"></i>#}
                                {#<p class="hidden-lg hidden-md">Search</p>#}
                            {#</a>#}
                        {#</li>#}
                    </ul>
                    <ul class="nav navbar-nav navbar-right">
                        <li>
                            <a href="{{ path('logout') }}">
                                <i class="ti-key"></i>
                                <p>Log out</p>
                            </a>
                        </li>
                    </ul>
                </div>
            </div>
        </nav>
    {% endblock %}

    {% block content %}

        <div class="container-fluid">
            <div class="card card-map">
                <div class="header">
                    <h4 class="title">Google Maps</h4>
                </div>

                <div class="map">
```

```
                    <div id="map"></div>
                </div>
            </div>
        </div>

        <div class="finalize-btn" style="display: none; position: absolute; bottom: 40px;
width: 100%; z-index:2000;">
            <div class="col-md-3"></div>
            <button class="btn col-md-6" style="border: transparent; font-family:
'Raleway';background-color: firebrick; color: white;text-transform: uppercase;letter-spacing:
1px; border-radius: 25px;">Finalize</button>
            <div class="col-md-3"></div>
        </div>


    {% endblock %}

    {#{% block footer %}#}
    {#{% endblock %}#}

    {% block modal %}

        <div id="myModal" class="modal fade" tabindex="-1" role="dialog" aria-
labelledby="myModalLabel"
             aria-hidden="true">
            <div class="modal-dialog">
                <div class="modal-content">
                    <div class="modal-header">
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">×</button>
                        <center>
                            <h3 id="myModalLabel">Create a New Job</h3>
                        </center>
                    </div>

                    <div class="modal-body" id="myWizard">

                        <div class="tab-content">

                            <div class="step-anim">
                                <div class="col-sm-12" style="z-index: 2">
                                    <div class="col-sm-4"></div>
                                        <p class="progress-steps col-sm-12" style="z-index: 2;
text-align: center;font-family: Raleway;font-size: 24px">Step 1 of 3</p>
                                    <div class="col-sm-4"></div>
                                </div>

                                <div class="progress col-sm-12"
                                     style="z-index: 2; background-color:
transparent;background: none;height: 20px;width: 100%;box-shadow: none">
                                    <div class="progress-bar col-sm-12" role="progressbar"
aria-valuenow="1" aria-valuemin="1"
                                         aria-valuemax="4"
                                         style="z-index: 2; width: 33%; border-radius: 25px;
background-color: rgba(127, 255, 127, 0.4)">
                                    </div>
                                </div>
                            </div>

                            <div class="tab-pane fade in active" id="step1">

                                <div class="card" style="margin-bottom: 0px !important;">
                                    <div class="header">
                                        <h4 class="title">General Information</h4>
                                    </div>
                                    <div class="content">
                                        <div class="row">
```

```
                                        <div class="col-md-12">
                                            <div class="form-group">
                                                <label>Title</label>
                                                <input id="job-title" type="text"
class="form-control" placeholder="Title">
                                            </div>
                                        </div>
                                    </div>

                                    <div class="row">
                                        <div class="col-md-12">
                                            <div class="form-group">
                                                <label>Employee</label>
                                                {#<input type="text" class="form-control"
placeholder="Employee">#}
                                                <input id="job-employee" list="employee"
type="text" placeholder="Employee"  class="form-control col-md-12">
                                                <datalist id="employee">

                                                    {% for user in users %}
                                                        <option value="{{ user.name }} {{
user.surname }}">
                                                    {% endfor %}

                                                </datalist>
                                            </div>
                                        </div>
                                    </div>

                                    <div class="row">
                                        <div class="col-md-6">
                                            <div class="form-group">
                                                <label>Deadline</label>
                                                <input id="job-deadline" type="date"
placeholder="Deadline" class="form-control">
                                            </div>
                                        </div>

                                        <div class="col-md-6">
                                            <div class="form-group">
                                                <label>Job Priority</label>
                                                <select class="form-control" id="job-
priority">
                                                    <option value="2">High</option>
                                                    <option value="1">Medium</option>
                                                    <option value="0">Low</option>
                                                </select>
                                            </div>
                                        </div>
                                    </div>


                                    <div class="row">
                                        <div class="col-md-6">
                                            <div class="form-group">
                                                <label>Reward</label>
                                                <input id="job-reward" min="0"
type="number" class="form-control" placeholder="Reward">
                                            </div>
                                        </div>
                                        <div class="col-md-6">
                                            <div class="form-group">
                                                <label>Job Difficulty</label>
                                                <select class="form-control" id="job-
difficulty">
                                                    <option value="0">Easy</option>
                                                    <option value="1">Challenging</option>
```

```
                                                    <option value="2">Hard</option>
                                                    <option value="3">Extreme</option>
                                                </select>
                                            </div>
                                        </div>
                                    </div>

                                    <div class="row">
                                        <div class="col-md-12">
                                            <div class="form-group">
                                                <label>Description</label>
                                                <textarea id="job-desc" rows="2"
style="resize:none" class="form-control" placeholder="Enter your description"></textarea>
                                            </div>
                                        </div>
                                    </div>
                                </div>
                            </div>
                            <div class="modal-footer">
                                <div class="col-md-1"></div>
                                <button class="btn col-md-5" aria-hidden="true" data-
dismiss="modal">Close</button>
                                <a href="#step2" class="btn col-md-5 btn-primary" data-
toggle="tab"
                                   data-step="2">Continue</a>
                                <div class="col-md-1"></div>
                            </div>
                        </div>

                        <div class="tab-pane fade" id="step2" >

                            <div class="card" style="margin-bottom: 0px !important;border:
transparent;box-shadow: none">
                                {#<div class="header">#}
                                    {#<h4 class="title">Dynamic Form</h4>#}
                                {#</div>#}
                                <div class="content">
                                    <div class="row">
                                        <div class="col-md-12">
                                            <div class="form-group">
                                                <div class="col-md-2"></div>
                                                {#<button class="btn col-md-8"
id="addAnotherSection" style="position: absolute;top:10px;left:70px">Add Section</button>#}
                                                <div class="col-md-2"></div>
                                            </div>
                                        </div>
                                    </div>


                                    <div class="row">
                                        <div class="flex-container">
                                            <div class="flex-item">
                                                <div id="mainContainer"></div>
                                            </div>
                                        </div>
                                    </div>

                                    <div class="row">
                                        <div class="form-group">
                                            <div class="col-md-5"></div>
                                             {#<button class="btn col-md-2"
id="addAnotherSection" style="text-align: center;margin-top: 10px" >Add</button>#}
                                            {#<button  class="btn btn-danger btn-circle
btn-xl col-md-4" style="  width: 60px;height: 60px;padding: 10px 16px;border-radius: 30px;font-
size: 24px;line-height: 1.33;"><i class="material-icons">add</i></button>#}
                                            <div class="col-md-5"></div>
                                        </div>
```

```
                                    </div>
                                </div>
                            </div>

                            <div class="modal-footer">
                                <div class="col-md-1"></div>
                                <a href="#step1" class="btn col-md-3" data-toggle="tab"
data-step="1">Back</a>
                                <a id="addAnotherSection" class="btn col-md-3 btn-
danger">Add Section</a>
                                {#<button class="btn col-md-2" id="addAnotherSection"
style="text-align: center;margin-top: 10px" >Add</button>#}
                                <a href="#step3" class="btn col-md-3 btn-primary" data-
toggle="tab"
                                   data-step="3">Continue</a>
                                <div class="col-md-1"></div>
                            </div>
                        </div>

                        <div class="tab-pane fade" id="step3">

                            <div class="card" style="margin-bottom: 0px !important;">
                                <div class="content">
                                    <div class="row">
                                        <div class="col-md-12">
                                            <div class="form-group">
                                                <label>Job Address</label>
                                                <input type="text" id="search-text-modal"
class="form-control" placeholder="Job Address">
                                            </div>
                                        </div>
                                    </div>
                                </div>
                            </div>
                            <div class="modal-footer">
                                <div class="col-md-1"></div>
                                <a href="#step2" class="btn col-md-5" data-toggle="tab"
data-step="2">Back</a>
                                <a href="" class="btn btn-danger col-md-5 search-btn-modal"
data-dismiss="modal" data-toggle="tab" data-step="3">Finish</a>

                                <div class="col-md-1"></div>
                            </div>
                        </div>

                    </div>
                </div>
            </div>
        </div>

    <div id="job-info-modal" class="modal fade" tabindex="-1" role="dialog" aria-
labelledby="myModalLabel"
         aria-hidden="true">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header">
                    <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">×</button>
                    <center>
                        <h3 id="myModalLabel">Job Information</h3>
                    </center>
                </div>

                <div class="modal-body" id="myWizard">

                    <div class="card" style="margin-bottom: 0px !important;">
```

```
                        <div class="header">
                            <h4 class="title">Job Information</h4>
                        </div>
                        <div class="content">

                            <div class="row">
                                <div class="col-md-12">
                                    <div class="form-group">
                                        <label>Title</label>
                                        <input id="info-job-title" type="text" class="form-
control" placeholder="Title" disabled>
                                    </div>
                                </div>
                            </div>

                            <div class="row">
                                <div class="col-md-12">
                                    <div class="form-group">
                                        <label>Employee</label>
                                        <input id="info-job-employee" type="text" class="form-
control" placeholder="Title" disabled>
                                    </div>
                                </div>
                            </div>

                            <div class="row">

                                <div class="col-md-6">
                                    <div class="form-group">
                                        <label>Deadline</label>
                                        <input id="info-job-deadline" type="text" class="form-
control" placeholder="Title" disabled>
                                    </div>
                                </div>

                                <div class="col-md-6">
                                    <div class="form-group">
                                        <label>Priority</label>
                                        <input id="info-job-priority" type="text" class="form-
control" placeholder="Title" disabled>
                                    </div>
                                </div>

                            </div>

                            <div class="row">

                                <div class="col-md-4">
                                    <div class="form-group">
                                        <label>Reward</label>
                                        <input id="info-job-reward" type="text" class="form-
control" placeholder="Title" disabled>
                                    </div>
                                </div>

                                <div class="col-md-4">
                                    <div class="form-group">
                                        <label>Difficulty</label>
                                        <input id="info-job-difficulty" type="text"
class="form-control" placeholder="Title" disabled>
                                    </div>
                                </div>

                                <div class="col-md-4">
                                    <div class="form-group">
                                        <label>Created At</label>
                                        <input id="info-job-created-at" type="text"
```

```
class="form-control" placeholder="Title" disabled>
                                    </div>
                                </div>

                            </div>

                            <div class="row">

                                <div class="col-md-12">
                                    <div class="form-group">
                                        <label>Description</label>
                                        <input id="info-job-description" type="text"
class="form-control" placeholder="Title" disabled>
                                    </div>
                                </div>
                            </div>
                        </div>

                        <div class="modal-footer">
                            <div class="col-md-1"></div>
                            <a class="btn col-md-5" data-dismiss="modal">Close</a>
                            <a class="btn col-md-5 btn-danger" id="delete-job">Delete Job</a>
                            <div class="col-md-1"></div>
                        </div>

                    </div>

                </div>
            </div>
        </div>
    </div>

    {% endblock %}


    {% block javascripts %}

        {{ parent() }}


        <script src="{{ asset('assets/js/map/map.js') }}"></script>

         {% for job in jobs %}
            <script>

                    var myMarker = new google.maps.LatLng({{ job.lat }},{{ job.lng  }});

                    var marker = new google.maps.Marker({
                     position: myMarker,
                     title: "Hello Tirana!"
                     });

                    marker.addListener('click', function() {
                        map.setZoom(8);
                        map.setCenter(marker.getPosition());
                        alert("Clicked on {{ job.title }}");
                    });

                     // To add the marker to the map, call setMap();
                     marker.setMap(map);

            </script>

         {% endfor %}


        <script>
```

```
$(document).ready(function () {

    $("#map-active").parent().parent().addClass("active");

    var allMarkers = [];
    var lastClicked;

    {% for job in jobs %}

        var icon;

        var priority = {{ job.priority }};

        if(priority === 0){
            icon = "assets/marker_icons/green_MarkerJ.png";
        }
        else if (priority === 1) {
            icon = "assets/marker_icons/orange_MarkerJ.png";
        }
        else {
            icon = "assets/marker_icons/red_MarkerJ.png";
        }

        var jobPosition = new google.maps.LatLng({{ job.lat }}, {{ job.lng }});

        var marker = new google.maps.Marker({
            position: jobPosition,
            title: '{{ job.title }}',
            icon : icon,
            map : map
        });

        marker.addListener('click', function () {

            var employeeName;

            map.setZoom(15);
            map.setCenter(new google.maps.LatLng({{ job.lat }}, {{ job.lng }}));
            lastClicked= '{{ job.id }}';

            var priority = {{ job.priority }};
            if(priority===0) priority="Low";
            else if(priority===1) priority="Medium";
            else if(priority===2) priority="High";

            var diff = {{ job.job_diff }};
            if(diff===0) diff="Easy";
            else if(diff===1) diff="Challenging";
            else if(diff===2) diff="Hard";
            else if(diff===3) diff="Extreme";

            $.ajax({
                type: "POST",
                url: '/get/employee',
                data: {
                    'id' : {{ job.user_id }}
                },
                success: function (msg) {
                    employeeName=msg;
                    $("#info-job-title").val('{{ job.title }}');
                    $("#info-job-description").val('{{ job.description}}');
                    $("#info-job-employee").val(employeeName);
                    $("#info-job-deadline").val('{{ job.deadline }}');
                    $("#info-job-priority").val(priority);
                    $("#info-job-reward").val('{{ job.reward }}');
                    $("#info-job-difficulty").val(diff);
```

```
                                $("#info-job-created-at").val('{{ job.created_at }}');
                                $('#job-info-modal').modal('show');
                            }
                        });
                    });

                    allMarkers.push(marker);

                    // console.log(marker);

                // marker.setMap(map);
                {% endfor %}


                $('#delete-job').click(function () {

                    $('#job-info-modal').modal('toggle');

                    $.ajax({
                        type: "POST",
                        url: '/delete/job',
                        data: {
                            'id' : lastClicked
                        },
                        success: function (msg) {
                            location.reload();
                        }
                    });

                });

                console.log(allMarkers);

                var markerCluster = new MarkerClusterer(map, allMarkers, {imagePath:
'https://developers.google.com/maps/documentation/javascript/examples/markerclusterer/m'});

            });
        </script>

    {% endblock %}
```

MEETLY TEAM

Deni Daja [ddaja15@epoka.edu.al**]**
Klesti Kuka[kkuka15@epoka.edu.al]
Bojken Sina[bsina15@epoka.edu.al]
Ernit Hoxha [ehoxha15@epoka.edu.al]