



CNG 491
SENIOR YEAR PROJECT: DESIGN

FIRST SPRINT REPORT

**Smart Database System
for
Disaster Recovery**

Contents

1	Introduction	4
1.1	What is Disaster Recovery System?	4
1.2	Why is this project being developed?	4
1.3	How is it being achieved?	5
2	System Requirements Specification	5
2.1	Functional Requirements	5
2.1.1	Web Application (WA)	5
2.1.2	Mobile Application (MA)	6
2.1.3	Wearable Application (WeA)	7
2.2	Non-Functional Requirements	7
2.2.1	Performance Requirements	7
2.2.2	Safety Requirements	8
2.2.3	Security Requirements	8
2.2.4	Software Quality Attributes	9
3	System Models	10
3.1	Use Case Diagram	10
3.1.1	Sequence Diagram	11
3.1.2	System Process Model	12
4	Scrum Details	13
4.1	Sprint Backlog	13
4.2	Sprint Burn-down Chart	14
4.3	Sprint Review	14

4.4	Sprint Retrospective	14
5	Project Estimation	15
6	Graphical User Interface (GUI)	19
6.1	Admin User (Web Application)	19
6.2	Normal User (Mobile Application)	21

Figures

1	Use Case Diagram for the Overall System	10
2	Sequence Diagram for the System with Normal User	11
3	Sequence Diagram for the Admin User	12
4	System Process Model	12
5	Sprint Backlog Table for DRS	13
6	Burn-down Chart of the Sprint	14
7	Project Characteristics	15
8	Functional Point Calculation	15
9	Lines of Codes (LOC) Calculation	16
10	COCOMO Results	16
11	Jones First Order Results	17
12	Schedule Estimation Results	17
13	Final Estimation with Three Different Methods	18
14	Compressing the Final Result	18
15	Admin User Login Page	19
16	Admin User Control Page	20
17	Normal User Login Page	21
18	Normal User Signup Page	22
19	Normal User Location Page (Only for Initial Prototype)	23
20	Normal User Operations Page	24

1 Introduction

This section provides an overview of the Distributed Database System for Location-based Control and Management System for Safety Warning and Emergency Rescuing Services (we will use Disaster Recovery System (DRS) as name of the project throughout the report) which gives answers to three main questions. First of all, DRS and its planned use will be explained as well as the main scope of the project and the reason behind developing DRS will be clarified. Lastly, details regarding the development process of DRS will be given with a particular explanation for each and every part of the project.

1.1 What is Disaster Recovery System?

DRS is a user location tracking system which will help rescue teams in case of an earthquake by providing the following information: After an earthquake, during the rescue process; rescue team will be able to know how many people there are under a wreckage of a specific building thanks to the fact that our software will send locations of the users and number of people around them. By getting this information, rescue team will choose the wreckage to rescue people by looking at the number of alive people under a wreckage. This will maximize the number of rescued people after an earthquake. For sending data to the servers we have several scenarios and they will be explained under the following sections.

1.2 Why is this project being developed?

As we all know, because of industrialization and job opportunities in big cities, people started to move from rural areas to big cities. This increased the population of the cities exponentially over the last years and because of the fact that cities have huge populations, the buildings started to get closer and closer to serve the purpose which increased the risk of being buried in a wreckage in case of an earthquake or being in a risky situation in case of a natural disaster. An example situation we can give to the things mentioned here is “Gölcük Earthquake” that occurred in 1999. In that earthquake nearly 17.000 people died and around 23.000 people was injured by being buried under a wreckage [1]. Another example which occurred in the near past is “Hurricane Irma” which caused 134 people’s death [2]. So, by developing such a system we are planning to optimize the rescue operation and decrease the number of deaths.

1.3 How is it being achieved?

There are several scenarios to achieve the planned functionality of the system. First thing that can be explained is how to get the location of the user. The system will have two options to get the location of the user: First one is Network Provider which gets the location from the cell tower that the user's phone is connected and the other one is GPS Provider which uses GPS sensor in user's phone to get the location of the user. When accuracy of two methods are compared, it can be seen that accuracy of the GPS sensor is higher. Although GPS's accuracy is higher, its power consumption is also high compared to the Network Provider. When it comes to passing the data to the server, several strategies will be applied at this point as well. The devices which have internet connections pass their data to the server and at the same time, they will be a gate for other devices which don't have internet connection. The ones with no connection will pass their data to the phone that is near them and then that data will be forwarded to the server. In the phones, we will have a mode called "Disaster Mode" in which the device will search for devices around it and try to find other people who are near. Also, in case of a disaster, the phone has to do some power saving in order to last longer under the wreckage and continuously pass data to our servers about whether the owner is alive or not. If the user has a smart watch connected to their mobile devices, the system will also get the heart rate of the user in that moment and continuously pass the live data to the server. The frequency for sending data to the server will be determined in the coming stages. Disaster Mode will also have the feature to switch the way to get the location of the user from GPS Provider to Network Provider in order to increase the battery life. In disaster mode, the device will try to find others using D2D connection since P2P connection might not be a good option.

2 System Requirements Specification

2.1 Functional Requirements

2.1.1 Web Application (WA)

1. Web application should have admin interface with exclusive login.
2. Web application should show the stored procedure options where "show user location", "get users at that location" can be given as examples, and these procedures can be executed via interface.
3. Web application should provide a web form for admin user panel, and a web service to establish connection between clients and the database server.

4. Web application should show the necessary inputs for the procedure. To illustrate, a procedure can take the credentials of the user to search and gather his/her nearby connections with the location information.
5. Selected procedures should be executed with given inputs, and the results should be shown in a grid, and also with Google Maps if applicable.
6. Web application should have a web service that is connected to the central database or databases in order to function as a mid-ware to handle requests from users (from mobile app) by manipulating the database.
7. Web application should be able to synchronize the data coming from different devices to create an overall map of the disaster area if the data sent by individual devices.
8. Between user and the database, web application should handle processes when needed in order to use less resources from user devices.
9. Web application should obtain and process the data from devices to generate a map, even if there is no disaster, to generate more accurate results.

2.1.2 Mobile Application (MA)

1. Mobile application should only allow authorized users to login.
2. Login page should provide user to login or the direct him/her to either password reset page or sign up page.
3. Mobile application should be able to use location information from either network provider or geolocation provider
4. Mobile application should be able to obtain the battery information of the phone.
5. Mobile application should be able to use wireless communication channels to either detect a possible disaster scenario or to find nearby devices.
6. Mobile application should be able to check and detect a possible disaster scenario based on the connectivity of the any available wireless network.
7. Mobile application should be able to convert the phone's state based on the detected disaster scenario for long battery life and required connections.
8. Mobile application should be able to store the nearby devices information in a local database.

9. Mobile application should be able to update its local database with specified frequency, if available, with the information gathered.
10. Mobile application should be able to create a D2D or P2P communication with nearby devices to understand how many people there are or to create a communication channel between them.
11. Mobile application should be able to connect to the web application and send the stored data.
12. Mobile application should be able to connect with a wearable which has the same application.
13. Mobile application should be able to gather the vital signs of the users from synced wearables' sensors such as heart rate sensor.
14. Mobile application should be able to transmit some information to the nearest device in a multi-hopping manner.
15. Mobile application should be able to transmit the data gathered to the web application for processing.

2.1.3 Wearable Application (WeA)

1. Wearable sensor should be able to sense heart rate.
2. Wearable be able to connect with the application in the mobile application
3. Wearable should be able to send its sensor data information to the web application via mobile application.

2.2 Non-Functional Requirements

This subsection presents the identified non-functional requirements for the project DRS. The subcategories of non-functional requirements given are performance, safety, security, software quality attributes and business roles.

2.2.1 Performance Requirements

Since human lives depend on the system, the performance is going to be a very important aspect. The system should provide accurate data.

1. System should be able to give an approximate location information with the optimized way considering both the power consumption and accuracy.
2. Web Application should be able to handle high number of requests without loss since disasters may affect large areas.
3. Mobile application should require less power to prevent battery draining in case of a disaster.
4. The local database should have sufficient memory for storing the latest k information. (where $k < 10$)
5. System should have a short response time to be able to work with large number of users.
6. Mobile application should control the maximum number of devices in a created group. In case of an overpopulation, new group may be needed.

2.2.2 Safety Requirements

1. System should update its content both locally and via web application even if there is no disaster to reduce the probability of undetected people in a disaster.
2. System should be able to restore itself in case of failure.
3. System should log the last date and time when the data obtained from mobile application.

2.2.3 Security Requirements

1. Every user should be registered to the system and should have a strong password.
2. There should be two types of users;
 - (a) Admin, with access to web application interface to use stored procedures.
 - (b) Normal user, with mobile app registration to use the service.
3. Centralized database(s) should only be called by the web application, and any other non-registered IP address will fail to connect to the database(s).

2.2.4 Software Quality Attributes

Reliability

As the concept of the project depends on the disaster scenarios, it is very crucial to have an up to date and accurate data, so that the actions can be taken accordingly. To achieve these goals, there needs to be some regulations about the intervals of the data gathering with most accurate and available methodologies, such as cell towers, GPS, network provider etc..

Another aspect that is important is to regulate the range between the users, so that word “close” stands for a standard for the project, and the decisions can be made easier when planning an action for the disaster situations.

For the management of these aspects, it is important to measure the accuracy of the methods so that a closeness definition can be made according to the range intervals, and for the timing of the data gathering, it is important to find the optimal solution where the power consumption of the user devices, server capacity and up to date data standards are considered.

Robustness

In case of a system failure, it is very critical to make sure that there will be no unwanted data generated by the system. In order not to lose information of data in case of system failure, all processing transactions in any situation should be rolled back. It should not take more than two minutes for the system to get started as the system is based on disaster recovery and even one second means a lot and decreasing the probability of data corruption should be prioritized.

Usability

The system should be designed and implemented in a way that it shows the desired data as clean as possible because under urgent situations the rescue team won't have time to understand the raw data. Also, error and help messages should be clear and understandable by anyone who reads it. Another point is graphical user interface; it should be clean and organized in a way that the thing user needs shouldn't be hard to find in software.

Maintainability

DRS must be designed and implemented with respect to the adaptability requirements of a possible upgrade for the software. Software upgrades may include expanding the functionality, improving the battery consumption and improving the GPS accuracy.

3 System Models

3.1 Use Case Diagram

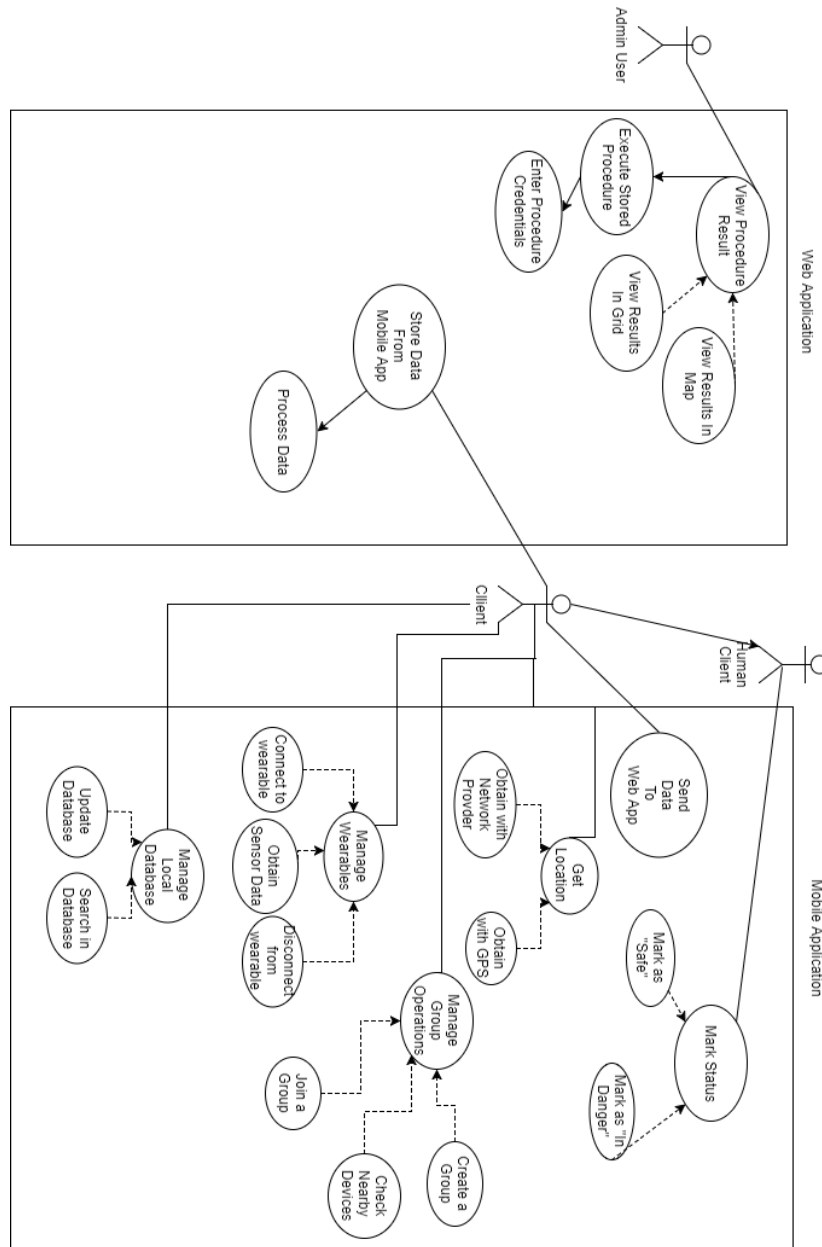


Figure 1: Use Case Diagram for the Overall System

3.1.1 Sequence Diagram

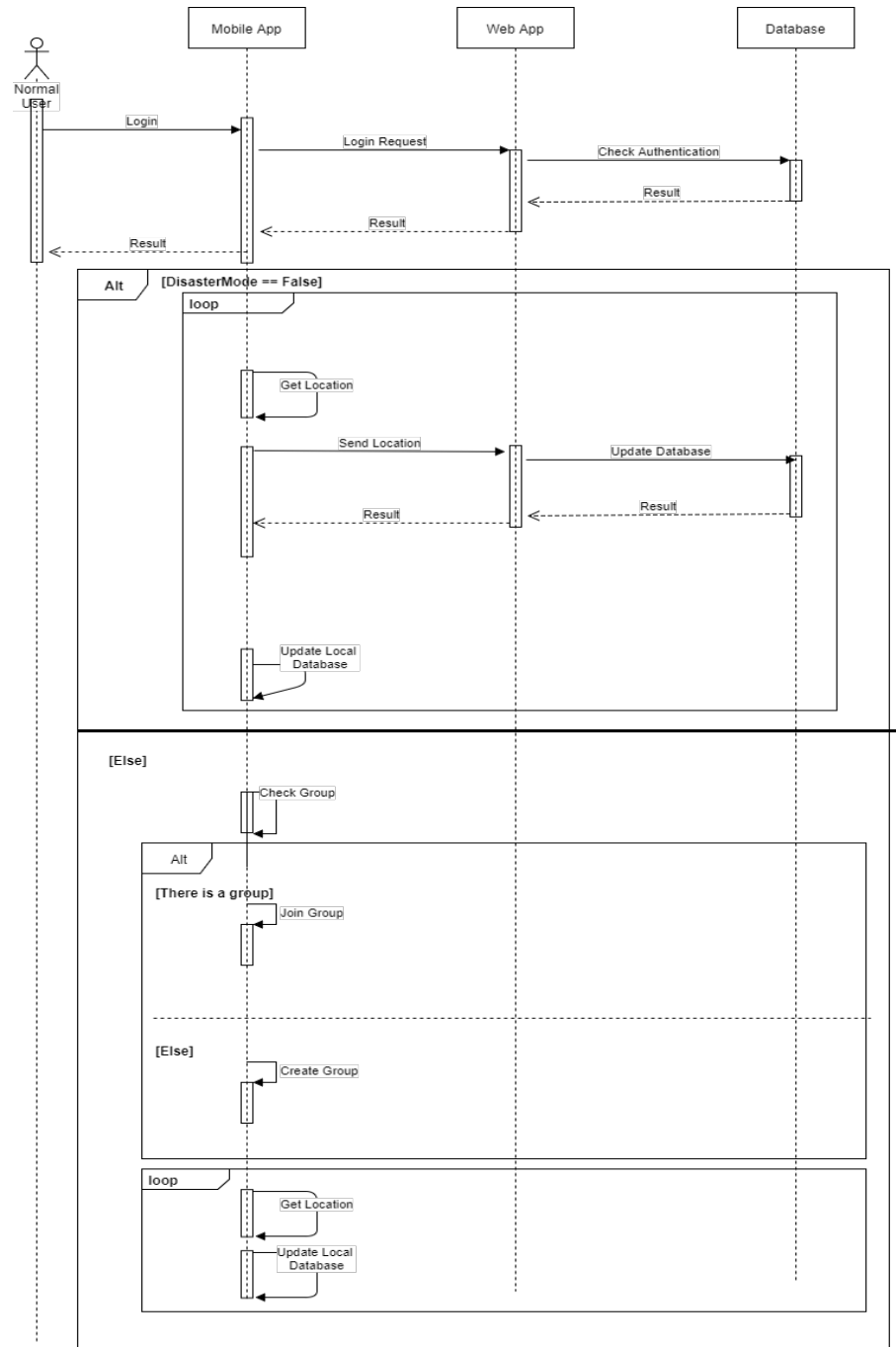


Figure 2: Sequence Diagram for the System with Normal User

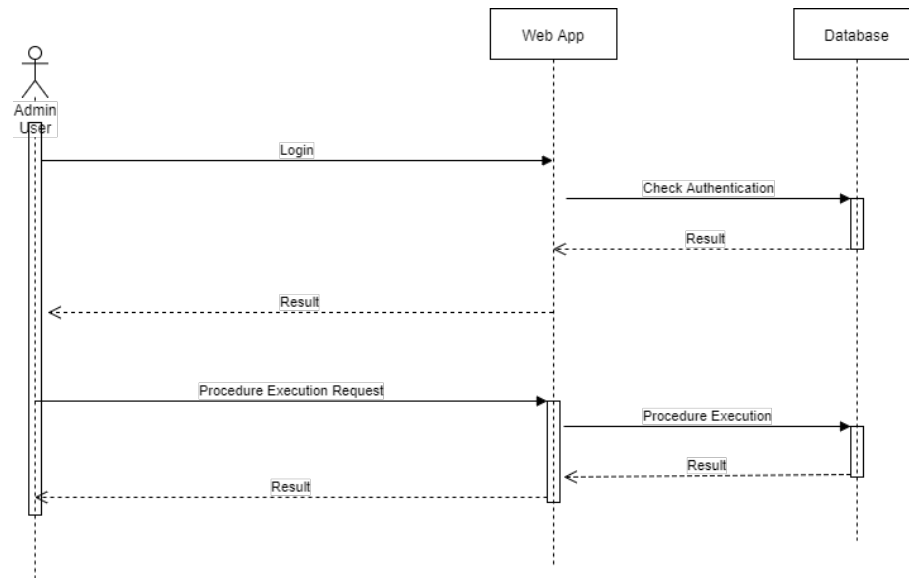


Figure 3: Sequence Diagram for the Admin User

3.1.2 System Process Model

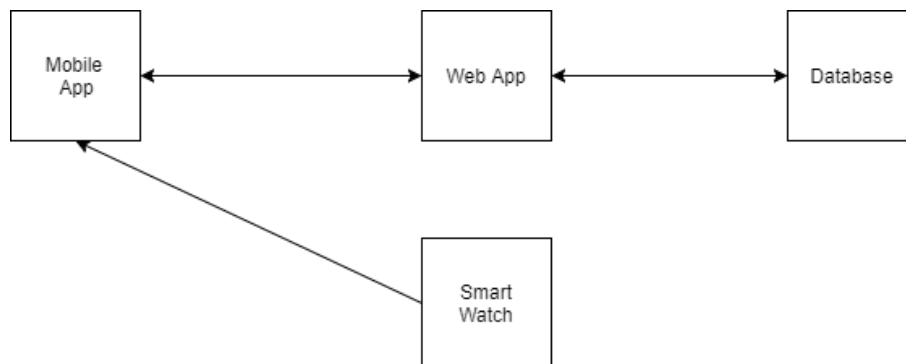


Figure 4: System Process Model

4 Scrum Details

This project was intended for two-three people. Since we have three people in our group, it was not that big deal to distribute the Sprint's tasks. In order to meet the deadlines, every member of the group tried to deal with the tasks as much as possible in their free time. For this sprint, we didn't have much difficulty in completing the tasks since they were some beginning tasks like researching, establishing server and application connection, login and signup screens of the application both functionally and visually. At the end of this sprint, we are now able to connect to the server from the application. We also created some mock user tables for logging in and signing up. After user logs into the system, s/he can see his/her own location from the application as latitude and longitude. In the second sprint, we will try to gather information about the devices that are around the user and combine that data with the location information to pass it to the server.

4.1 Sprint Backlog

SPRINT 1 (01.10.2017 - 31.10.17)	
Description	Effort in Hours
Learning about the existing infrastructure being used	5
Creating the model of the system	3
Creation and deployment of the server	2
Installing the Android Studio and necessary SDKs to develop the App	1.5
Writing code for backend of the system	1
Research about the Network Topologies	2
Writing Login and Signup Portal of the System	1
Establishing App - Server Connection	6
Getting User's Location with GPS and Network Provider	3
Writing a service to get user's location at the back continuously	0.5
Creating a P2P Service discovery mechanism for initial prototype	4

Figure 5: Sprint Backlog Table for DRS

4.2 Sprint Burn-down Chart

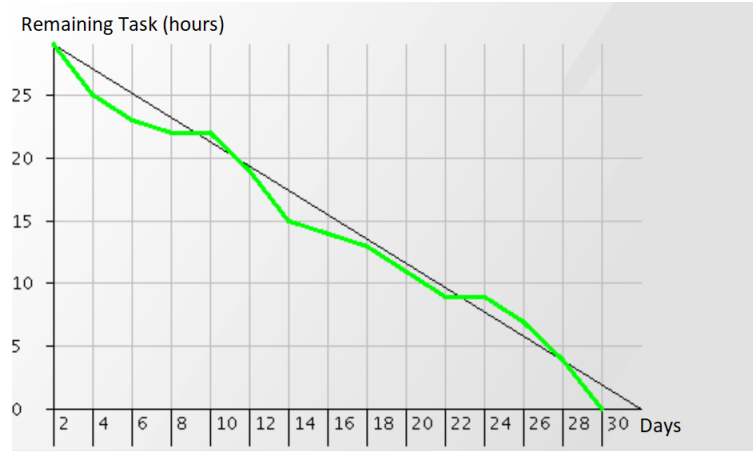


Figure 6: Burn-down Chart of the Sprint

4.3 Sprint Review

While finishing the first sprint, we realized that we mainly focused on to understand the basics of the project i.e. the requirements and to build a simple prototype based on these requirements to understand for the second spring more. As can be seen in the sprint backlog and the burn-down chart, we spend more time on to build a concrete application than researching. In this process, we had a problem with the application-server connection which makes us frustrated in the middle of the sprint. Moreover, we have come to the conclusion that we try to change the requirements which is conflicting with the sprint rules. Hence, during this sprint we understand more about the scrum as well as our requirements. All in all, we believe that after this sprint we will be able to proceed more actively and efficiently by using the ideology of the scrum more.

4.4 Sprint Retrospective

For the retrospective i.e. the team evaluation itself, we can say that we panicked at first because of the fact that the overall concept was new for the team. Due to this, we weren't able to create an appropriate product in the sprint. Moreover, the understanding of the requirements varied among the team during the sprint which makes every member to expect and to build different things from the agreed components for the prototype. Despite the ini-

tial chaotic environment, we achieved to proceed and to build the simple parts of the prototype while understanding the required background information about the project. To conclude, we are planning to be more organized and less panicked in order to enhance our prototype to the desired state.

5 Project Estimation

STEP 1 : PRODUCT CHARACTERISTICS			
General Systems Cchs (GSC) - FPA	Value	General Systems Cchs (GSC) - FPA	Value
1- data communications	5	8- online updates	4
2- Distributed Functions	5	9- complex processing	3
3- Performance	5	10- reusability	3
4- Heavily used configurations	5	11- installation ease	1
5- Transaction Rate	5	12- operations ease	2
6- online data entry	2	13- Multiple sites	1
7- end user efficiency	1	14- Facilitate change	2
Total:	44	IM/VAF	1.09

Figure 7: Project Characteristics

Char	Low	medium	high	Low	Medium	High
input	4	2	2	12	8	12
output	5	0	2	20	0	14
inquiries	6	0	1	18	0	6
logical internal	2	5	0	14	50	0
external interface	0	1	0	0	7	0
Unadjusted Total of Function Points	161			64	65	32
Adjusted Total of Function Points	175.49			Influence Multiplier (IM)		1.09

Figure 8: Functional Point Calculation

Language	Minimum	Moderate	Maximum
C	10529.4	22462.72	29833.3
C #	7019.6	9651.95	14039.2
C++	7019.6	9651.95	24568.6
Java	7019.6	9651.95	14039.2
SQL	1228.43	2281.37	2632.35

Figure 9: Lines of Codes (LOC) Calculation

For the LOC, we selected Java and C. However, as they have the same lines of code value in average class, we did not calculate any mean.

Cocomo Basic	KDSI (KLOC)	Man Month	TDEV
Organic	9.65195	25.94520212	8.615523886
Semi-Detached	9.65195	38.00932226	8.931030093
Embedded	9.65195	54.68151983	8.995990607

Figure 10: COCOMO Results

For the COCOMO, we selected the Semi-Detached project type where the formula for the man month is,

$$MM = a \times KDSI^b \text{ (where } a = 3.0, b = 1.05 \text{ } KDSI = 9.65195) \approx 38 \quad (1)$$

By looking at the COCOMO results, project requires 38 man-month. Since our group contains 3 people who interested in different parts of the project, the workload will be equally distributed. By checking with our team size,

$$TDEV = \frac{ManMonth}{\#ofWorkers} = \frac{38}{3} \approx 13 \text{ months} \quad (2)$$

Hence, the project is doable based on COCOMO results with a little compression. However, we'll continue the estimation to obtain a more accurate result.

Jones's First-Order Effort Estimation			
Kind	Best	Average	Worst
Systems	29.08853159	39.66221283	63.14800786
Bussiness	21.3337232	29.08853159	46.3131704
Shrink Wrap	15.64629497	24.91117583	39.66221283

Figure 11: Jones First Order Results

For the Jones First Order, we selected the System class which gave us the average 29 man months. Since our team is consists of 3 people, the estimated development time is,

$$TDEV = \frac{ManMonth}{\#ofWorkers} = \frac{29}{3} \approx 10 months \quad (3)$$

As a last method, we used the Schedule Estimation method by assuming that we are in average developers scope.

STEP 4 : SCHEDULE ESTIMATION - Efficient Schedule						
Lines of Code	System Products		Business Product		Shrink-Wrap Products	
	Schedule(Months)	Man-Month	Schedule(Months)	Man-Month	Schedule(Month)	Man-Month
10,000	8	24	4.9	5	5.9	8
15,000	10	38	5.8	8	7	12
20,000	11	54	7	11	8	18
25,000	12	70	7	14	9	23
30,000	13	97	8	20	9	32
35,000	14	12	8	24	10	39
40,000	15	140	9	30	10	49
45,000	16	170	9	34	11	57
50,000	16	190	10	40	11	67
60,000	18	240	10	49	12	83
70,000	19	290	11	61	13	100
80,000	20	35	12	71	14	120
90,000	21	400	12	82	15	140
100,000	22	450	13	93	15	160
120,000	23	560	14	115	16	195
140,000	25	670	15	140	17	235
160,000	26	709	15	160	18	280
180,000	28	910	16	190	19	320
200,000	29	1300	17	210	20	360
250,000	32	1300	19	280	22	470
300,000	34	1650	20	345	24	590
400,000	38	2350	22	490	27	830
500,000	42	3100	25	640	29	1100

Figure 12: Schedule Estimation Results

The estimated man month is 24. Again using the formula,

$$TDEV = \frac{ManMonth}{\#ofWorkers} = \frac{24}{3} \approx 8 months \quad (4)$$

To obtain an average estimation for man month, we used the formula,

$$Expected = \frac{2 \times Worst + 3 \times MostLikely + Best}{6} \approx 36 \quad (5)$$

where

- Best = 24 MM
- Most Likely = 38.00932226 MM
- Worst = 39.66221283 MM

$$Standart\ Deviation = \frac{Worst - Best}{6} = 2.61 \quad (6)$$

All in all, when we combined all these information and created the table for final estimations with the team size 3, the result is,

Enter Team Size		>>>>>>	3					
Phase	Effort & Size -Man Month							
	Optimistic	Pessimistic	Days		Days			
Initial product concept	2.873037637	45.96860219	86.1911291		1379.058066			
Approved product concept	5.746075273	22.98430109	172.3822582		689.5290328			
Requirements Specification	7.699740866	17.23822582	230.992226		517.1467746			
Product Design Specification	9.193720437	14.36518818	275.8116131		430.9556455			
Detailed Design Specification	10.34293549	12.6413656	310.2880648		379.240968			
Above is the interval					Above is the interval			

Figure 13: Final Estimation with Three Different Methods

Hence, even though we obtain different results from different estimation methods, the overall estimation is between 10-12 months which is doable with a small compression which makes the range between 8-10 months in the figure below.

Enter Compress Rate	>>>>>>	0.8	>>> Warning : Below 0,75 is NOT recommended. Can be used to co			
			Days		Days	
			68.95290328		1103.246452	
			137.9058066		551.6232262	
			184.7937808		413.7174197	
			220.6492905		344.7645164	
			248.2304518		303.3927744	

Figure 14: Compressing the Final Result

6 Graphical User Interface (GUI)

6.1 Admin User (Web Application)



The image shows a web application login page. At the top, there is a black banner with the text "DISASTER STRIKES" in large, white, distressed font. Below the banner, the login form is centered. It consists of two input fields: "User ID" and "Password". The "User ID" field contains the text "METUNCCSmartDB" and has a dropdown arrow icon to its right. The "Password" field contains ten asterisks "*****" and has a refresh/clear icon to its right. Below these fields is a "Log in" button with a lock icon. At the bottom of the page, there is a decorative horizontal bar with a black top half and a red bottom half.

DISASTER STRIKES

User ID : 

Password : 




Figure 15: Admin User Login Page

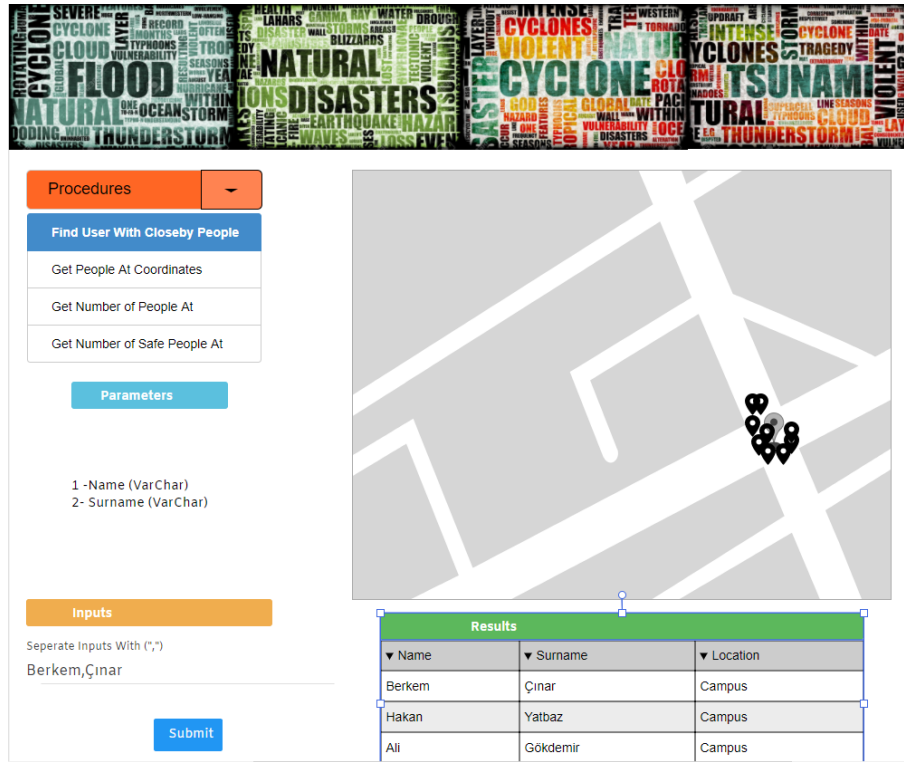
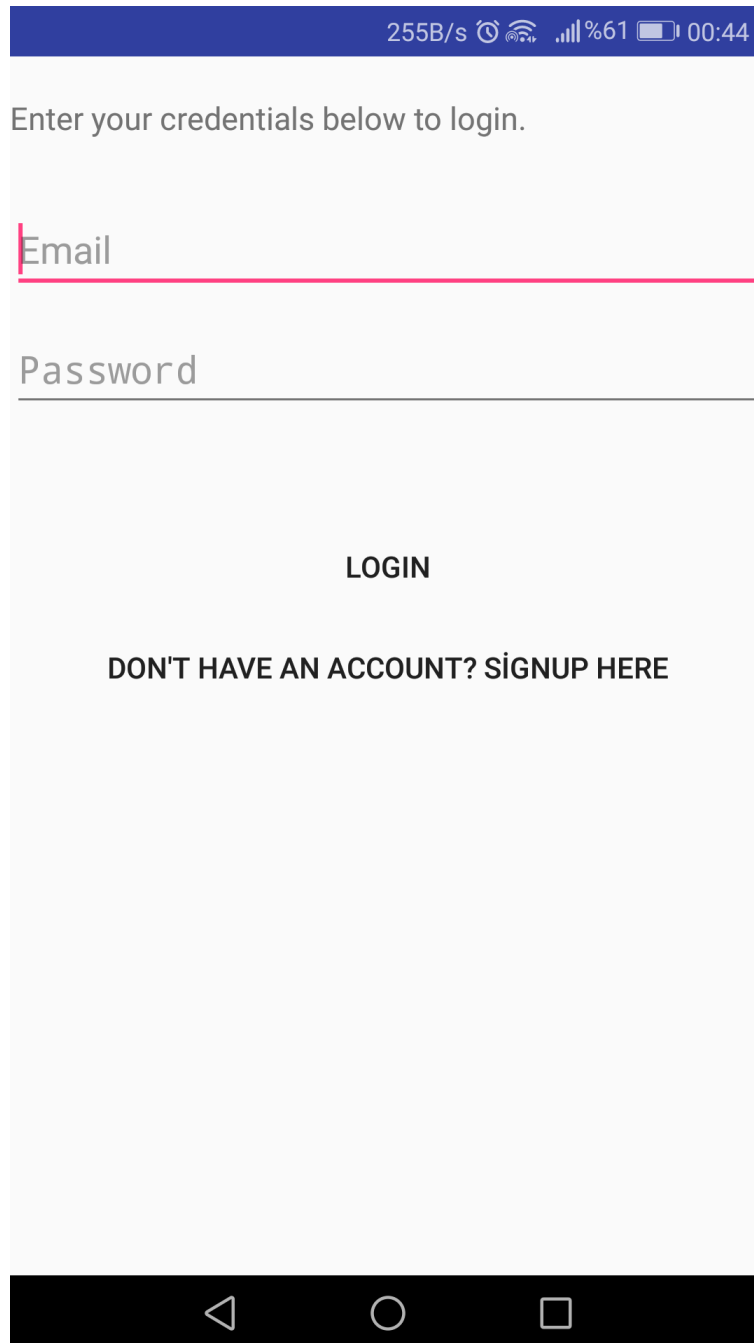


Figure 16: Admin User Control Page

As seen in the figures, the assigned admin will be able to enter, login, the web application side of the system to query the database by using the written procedures for simplicity and visualizing the results in either in map or in a list.



6.2 Normal User (Mobile Application)





A screenshot of a mobile application login page. At the top is a dark blue status bar with white text and icons: '255B/s', a clock icon, a Wi-Fi icon, a cellular signal icon, a battery icon at 61%, and the time '00:44'. Below the status bar is a light gray background. The text 'Enter your credentials below to login.' is displayed in a dark gray font. There are two input fields: the first is labeled 'Email' with a red vertical line on its left side, and the second is labeled 'Password'. Below the input fields, the word 'LOGIN' is centered in bold black text. Further down, the text 'DON'T HAVE AN ACCOUNT? SIGNUP HERE' is centered in bold black text. At the bottom of the screen is a black navigation bar with three white icons: a back arrow, a circle, and a square.

Figure 17: Normal User Login Page

262B/s

 %61

 00:44

Please enter the credentials below to signup.

Name

Email

Password

SIGNUP

Figure 18: Normal User Signup Page

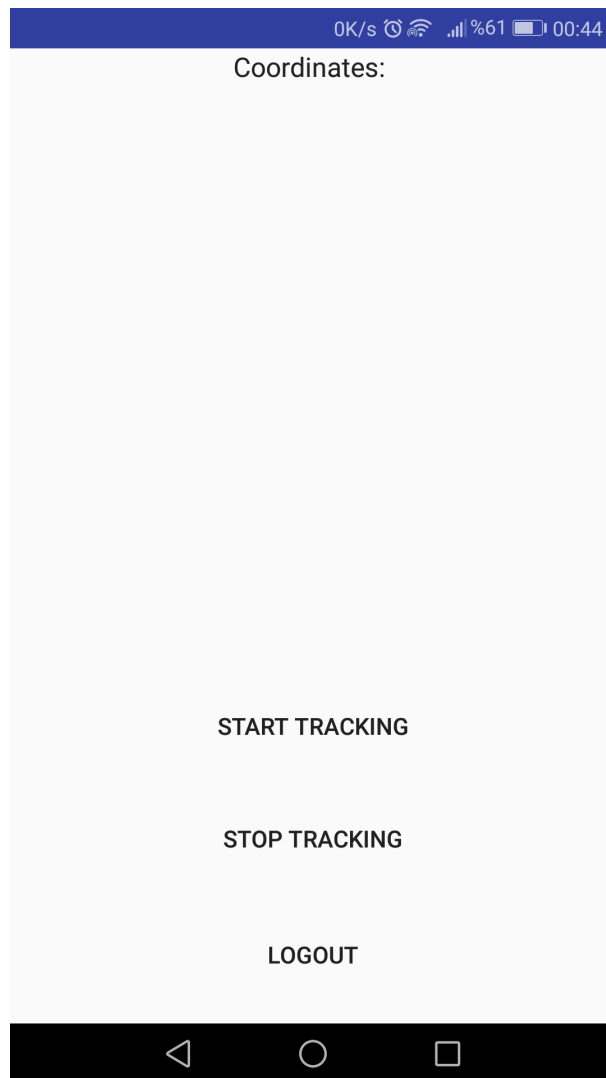


Figure 19: Normal User Location Page (Only for Initial Prototype)

The last page is only for debugging and testing purposes. It won't be occur in the final design. However, instead of this page an activity for syncing the smart watches and marking oneself as safe option will be included in the final design. Initial design is provided below.

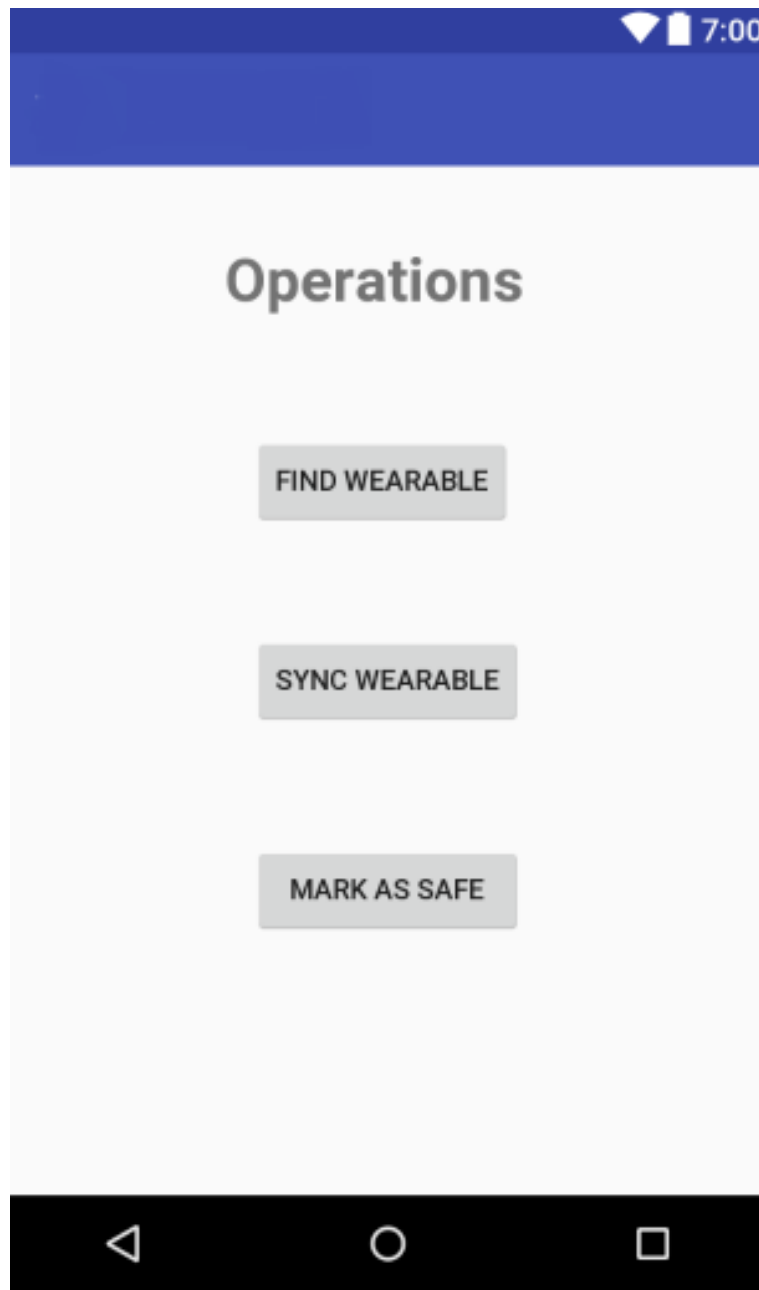


Figure 20: Normal User Operations Page

References

- [1] *1999 İzmit earthquake*. [Online]. Available:
<https://en.wikipedia.org/wiki/1999>
- [2] *Hurricane Irma* [Online]. Available:
https://en.wikipedia.org/wiki/Hurricane_Irma [Accessed : 01 – Nov – 2017].