Software Requirements Specification

for

Disaster Management Application

Version 2.0 approved

Prepared By:

Gaurav U D (16IT113)

Shrinivas V Shanbhag (16IT244)

Abhilash V (16IT201)

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The reason for this report is to introduce the most practical details of the Disaster Management Application. It will clarify the reason and highlights of the application, the interfaces of the application, what the application will do, the limitations under which it must work and how the application will respond to outer upgrades. This archive is proposed for both the partners and the engineers of the framework.

1.2 Document Conventions

Main heading - Font size 18
Sub heading - Font size 14
Paragraph - Font size 12

Abbreviations used:
API :- Application Programming Interface

1.3 Intended Audience and Reading Suggestions

This document is intended for the developers and the testers for understanding the requirements of the user. The user can also check if his requirements are satisfied with the help of this document.

1.4 Product Scope

This project is intended for the use of general public, relief workers, government and any other entities affected by a natural disaster or any other organization who are willing to extend their help towards the affected.It is software which provides a platform to swiftly mitigate and manage any disaster by connecting key players - donors,volunteers/rescue teams and NGOs at one place. This software can be easily used to create disaster specific solution with disaster specific details updation by basic developer and then it is up and running for use to manage the disaster.
The basic aim of the system is to manage a disaster efficiently and warn people about dangers and at the same time help the people those who are in danger.

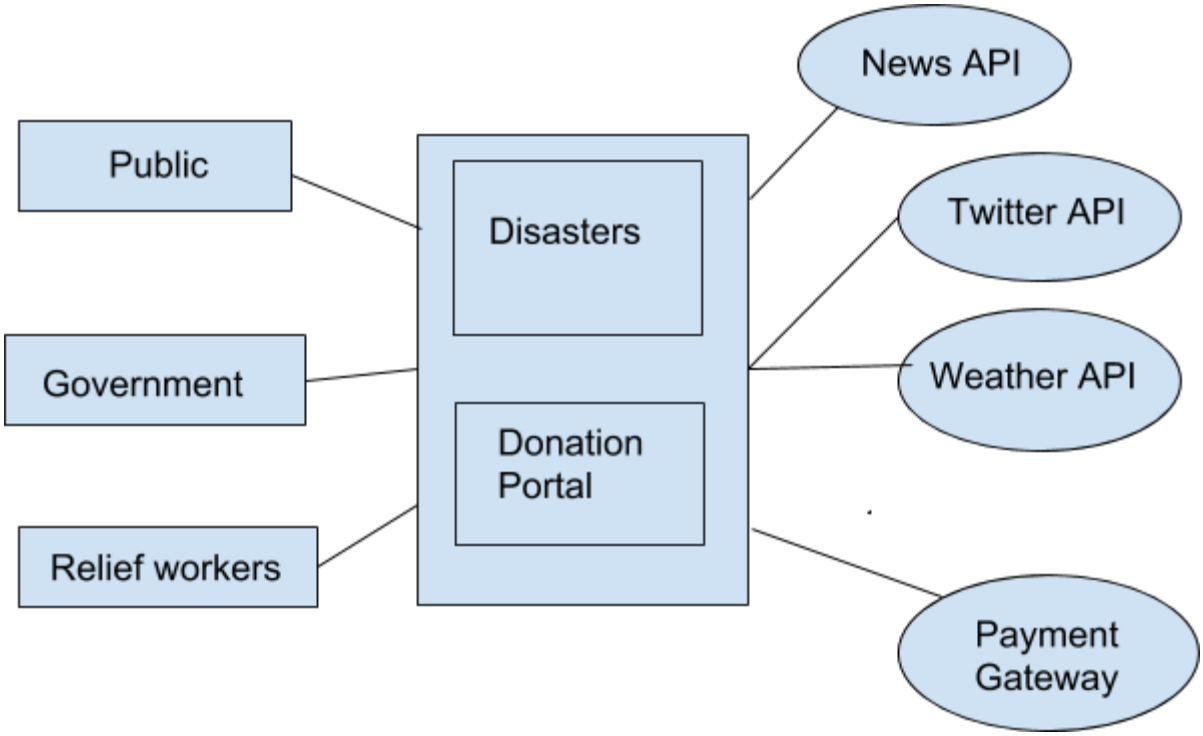
1.5 References

API documentations provided by various other Web Interfaces such as PredictHQ, Twitter, News API, OpenWeather API.

2. Overall Description

2.1 Product Perspective

This product is service which is built on many other microservices such as Google APIs, Twitter APIs, News APIs, Weather APIs, Payment gateway. Hence the functionality of this product depends on the functionality of all these microservices.



2.2 Product Functions

This application has functions like listing all disasters, marking the safe and dangerous locations in the map where the disaster is occuring, donation portal for gathering funds for a particular disaster. This application also shows News, Tweets, Weather of that particular location where the disaster is occuring.

2.3 User Classes and Characteristics

Different Users who will be using this application are:

- *Public:*
  Here the registered users can add new markers to the map pertaining to a particular disaster. The unregistered user can just view the maps without being able to make any changes
- *Government:*
  Government of the affected region can use the application to find out which are the areas which require more resources
- *Relief Workers:*
  More workers can move towards places which need more help by using the applications
- *Donors:*
  People can donate money for the people affected by these disasters.

2.4 Operating Environment

This product has been planned to be dockerized for easy deployment on any servers such as AWS or Azure. This project is to be built in Python using Django framework.

2.5 Design and Implementation Constraints

There might be a possibility that the API results may not be filtered according to our search requirements.

2.6 User Documentation

The precise documentation for this website should be made available in the help section of the website.

2.7 Assumptions and Dependencies

This project heavily depends on APIs provided by many other third party services and it is the assumed that these services will be operational and operate as expected throughout the duration of this project.

3. External Interface Requirements

3.1 User Interfaces

The user will interact with this application through a website which is built using Django. There will different pages for different disasters and each disaster will have its own dedicated map and other important features.

3.2 Hardware Interfaces

The application will be deployed on Azure.

3.3 Software Interfaces

SQL Lite database is used for the storing the data relating to this application. Django has in built modules to communicate with the database and storing data

3.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

4. System Features

4.1 Google maps

4.1.1 Description and Priority

This has the highest priority in this project.

Here there will be a map provided of the area where the disaster is occuring.

The users have to mark which locations are safe, dangerous and where help is needed the most.

4.1.2 Stimulus/Response Sequences

Drag and drop pins of different colors to represent different categories will be provided to mark the areas on the map. The user must be logged in in order to use this functionality

4.1.3 Functional Requirements

REQ-1: Google Map API

In case the user who has not logged in is trying to drop a marker, he will be redirected to the login page

4.2 Donation Portal

4.2.1 Description and Priority

This has high priority in this project.

Here there will be a payment gateway for for making donations to a particular disaster

The donors can donate a certain amount to the relief workers who are working on a particular disaster.

4.2.2 Stimulus/Response Sequences

The donor will be redirected to the payment gateway where the donor has to complete his payment

4.2.3 Functional Requirements

REQ-1: Bank account details of donor
REQ-2: Bank account details of receiver

Both the donor and receiver must have registered in the application along with their Bank details.

4.3 PredictHQ API

4.3.1 Description and Priority

This has high priority in this project.

PredictHQ API provides exact details of the disasters that are currently occurring throughout the world.

4.3.2 Stimulus/Response Sequences

The user has to select a disaster from the list of disasters provided by this API

4.3.3 Functional Requirements

REQ-1: PredictHQ API key

4.4 News API

4.3.1 Description and Priority

This has medium priority in this project.

News API provides news articles relating to the disasters.

4.3.2 Stimulus/Response Sequences

User can learn more about the recent happenings in the disaster area.

4.3.3 Functional Requirements

REQ-1: News API key

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The application has to run on areas where the internet connectivity is very less and hence the website must be light and not have too much redundant data. It should also be optimized to run efficiently on mobile and other handheld devices.

5.2 Safety Requirements

Since all the data which has been posted on this website are from the users, there is no guarantee of the authenticity of the data and the application is not responsible for any fake data which has been posted.

5.3 Security Requirements

The bank details must be stored securely in the database and there must not be any data leaks.

5.4 Software Quality Attributes

The website must have these qualities: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability.

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>