Technical Documentation

Incidentz – 2017
[Mobile Application]

Yanik Blake 1400143100
Tarj Davy 1400034985

Name of the project	Incidentz
Type of document	Technical Document
Date	16/06/2017
Version	1
Authors	Tarj Davy Yanik Blake
Link	https://github.com/Yanik-Dev/incident_app

Modification

Version	Date	Names	Description

Table of Contents

Introduction

This section gives the scope and overview of everything included in this Technical document. The purpose for this document is described and a list of abbreviations and jargon's definition is provided.

Purpose

The purpose of this document is to give a detailed description of the requirements for the application, “Incidentz”. It will illustrate the purpose and complete declaration for the development of the system. It will also explain the system constraints, interface and interaction between system components.

Definitions, acronyms, and abbreviations

Terms	**Definitions**
MVVM	Model View View Model
Frontend	The front-end is everything involved with what the user sees and directly interacts with.
Backend	This refers to everything the user can't see like databases and servers.
JSON	JavaScript Object Notation
Framework	A framework is a concrete or conceptual platform where common code with generic functionality can be selectively specialized or overridden by developers or users
GPS	Global Positioning System
Database	A computer software that is used to store data
Incident	An incident is defined as anything that may affect land commuters from traveling safely to their location. An incident maybe but not limited to any of the following: 1. Road Block 2. Car crash 3. Flooding 4. Oil Spill 5. Forest Fire 6. Road Break away 7. Landslide 8. Heavy Rain 9. Police on duty
Stakeholder	Any person who interacts with the mobile application and is not a developer
Geo-Location	
NoSQL	No Structure Query Language A branch of database that does not adhere to a predefined structure.
API	Application Programming Interface
UI	User Interface Describe the look and feel of the system
UX	User Experience Deal with how responsive the mobile application is to user’s interaction

Problem Statement

The development team have observed that while traveling incidents can occur anywhere at any point in time. Whether it be as simple as a roadblock or as serious as a flood, most commuters are unaware that such incidents have occurred and as a result get caught up in these incidents. Getting caught up in these incidents could then result in lost of valuable time, when an alternate route could have been taken; Traffic jams, for example the more persons not aware of a roadblock happen at X may help contribute to the traffic jam; Injure and possible lost of life due to life threatening incidents. All those could have been avoided if commuters were aware of the incident.

Proposed Solution

Incidentz is a mobile application focusing on aiding the commuters to get notified before getting caught up in an incident on the road.

We hope to create a system where commuters no longer have to get caught up in an incident to be aware of what is going on but instead with as simple as a notification they are aware that an incident as occurred. With the help of first-on-scene commuters, By posting an image of the incident accompanied by their location the app will be able to notify other commuters using the app that their is an incident at that specific location.

Affected Industry	All
Stakeholders	Commuters

Product Perspective

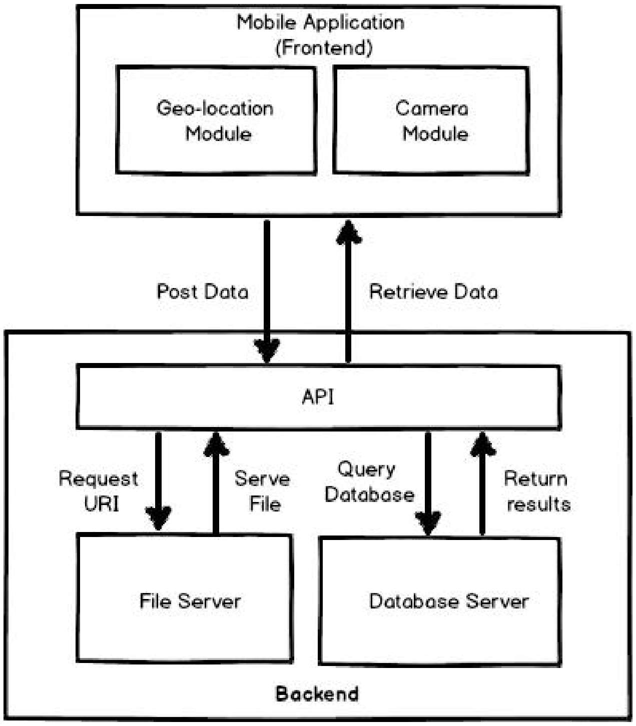


Figure 1: Block diagram of the system.

This system consist of two parts: One front-end mobile application and an API (Back-end). The mobile application will be used to post and view incidents by users whereas the API will be used to provide a gateway for the mobile application to interacts with the database as seen in Figure 1.

The mobile application will use Geo-location coupled with the Camera Module in order to post details of an incident to the system. The GPS from the Geo-location module will provide the users current location coordinates. The Camera module will enable users to post images of the incidents. Both the coordinates from the Geo-location and an image from the camera module is required to post to the API. All post will be displayed on the mobile application on a map using an icon depicting the incident and lost as long as one day unless deleted by the person who posted the incident. The Geo-location module will also be used to get the users coordinates in order to show users all the incidents in and around their location.

Since the mobile application is a data-centric product, A database as shown in Figure 1 will be used to store and retrieved textual information. Files such as images will be stored on file storage server as shown in Figure 1. The mobile application will only be able to interact with the database through the API. The API will consist of defining how data should be structured to and from the database. All API communication will go over the internet.

Product Requirements

The “Incidentz” mobile application is GPS-based mobile application which aid as a notification system where commuters can notify other commuters that there is an incident nearby. The App will mainly use the Camera and Geo-location feature on any of the three (3) mainly used mobile operating system. The mobile application should carry out the following requirements:

Functional Requirement

- Take images of Incidents. (Users are not allowed to upload images from their photo album)
- View map with markers of all post. Each marker should have an icon depicting the specific incident.
- On click of the markers, a pop up displaying the image and a comment from the user who posted it
- View cards with markers of all post. Each card should have image(s) of incident, icon depicting incident, user who post the event along with time and date.
- Each user should have a history of all their posts.
- Push notification of important alerts.
- Notify user if user is close to an incident

Non-Functional Requirement

The success of this project also depends on the following non functional requirements.

- Usability
  The mobile application should be easy to adapt to. The mobile application should be

- Privacy
  The mobile and web application should be developed in such a way that users information is kept securely.

- Reliability and Availability
  The mobile application should always be working once there is connection to the internet

- Scalability
  The mobile application should be designed to accommodate new features.

- Integrity

  Data contents and structures should be preserved when failures occur.

Native Functions

- Geo-location
  This native functionality will be used when user post an incident to capture the coordinates of the user's location. The coordinates will then be use to map each incidents on the map. Additionally it will also be used to get the user's current location to enable users to see all the incidents in their location.
- Camera

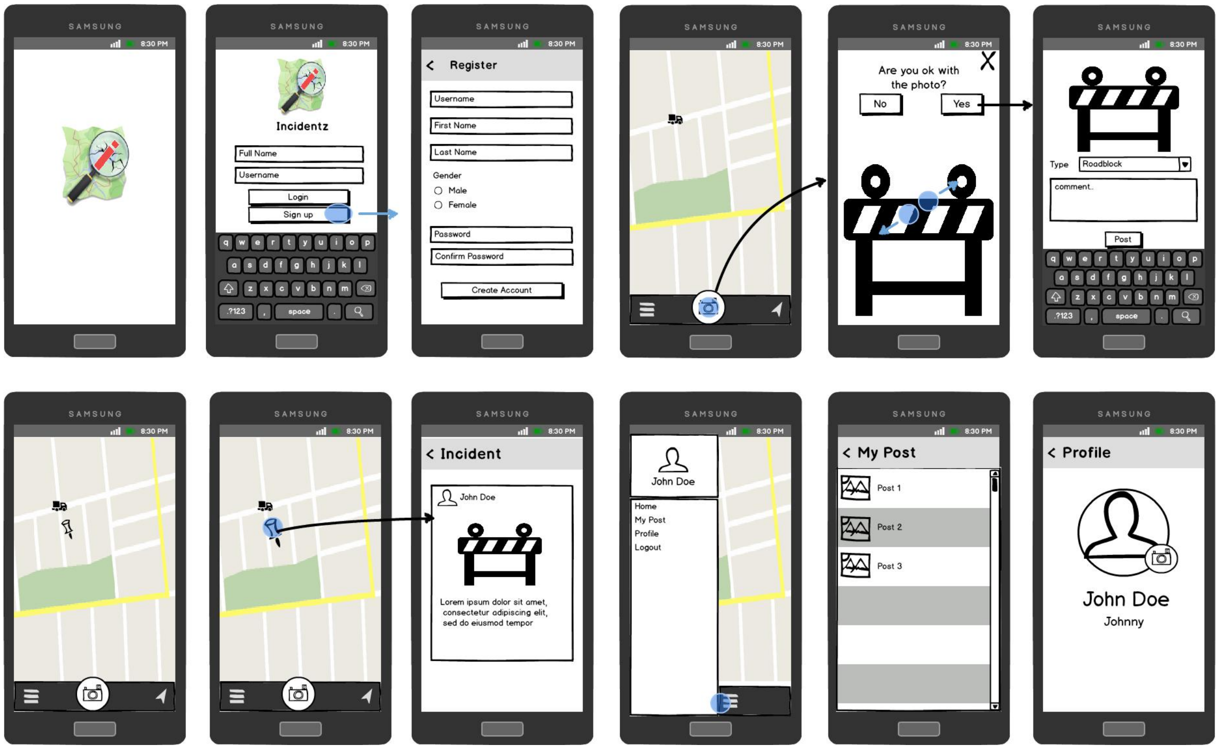
  This Native feature will be use to capture an image of the incidents and also give access to the user's gallery to enable users to change their profile picture.
- Network Connection

  This feature will give the application access to the device's network connection state enabling the app to make decision if the application is offline or online.

Technologies & Software Used

- Ionic 3 Framework - (Mobile framework built using the angular 4 framework)
- Angular 4 - (A JavaScript MVVM framework used to development frontend application)
- Firebase - (A real time NoSQL database used to store records in a JSON document oriented format)
- Node JS - (A JavaScript runtime environment)
- Visual Studio Code - (Text editor)
- Ionic Cloud and Github (Version control and Collaboration)

UI Mock Up



UI Design Rational and Justification

Colours : Matte Black, Gray, White
These colours were used throughout the app to promote readability in the night. When the user move his/her eyes from the dark theme on the app to the darkness at night the user's eyes wont need to adjust significantly as when moving his/her eyes from a light theme.

Buttons: Standard mobile size material buttons will be used across all mobile platforms to facilitate user experience.Users will be able to tap any button without conflict with any other elements on the screen.

Content Spacing : 20px padding within all container elements to enhance readability and to utilizing space efficiently.

Navigation:
<u>Side Menu</u>
This will be use enable users to easily navigate to unrelated content within the application as this feature is a standard feature in many popular apps so users should have no problem locating and using this feature.
<u>Tabs</u>
This will be located at the bottom of the mobile device screen on the home page/view to enable easy access to related content. All items on the tab will be frequently used features of the application.

Database Diagram

Database was developed using a NoSQL structure therefore the structure of the database does not truly reflect the database diagram as the NoSQL database uses a denormalization structure. However the Diagram was used as a guideline in constructing the database.

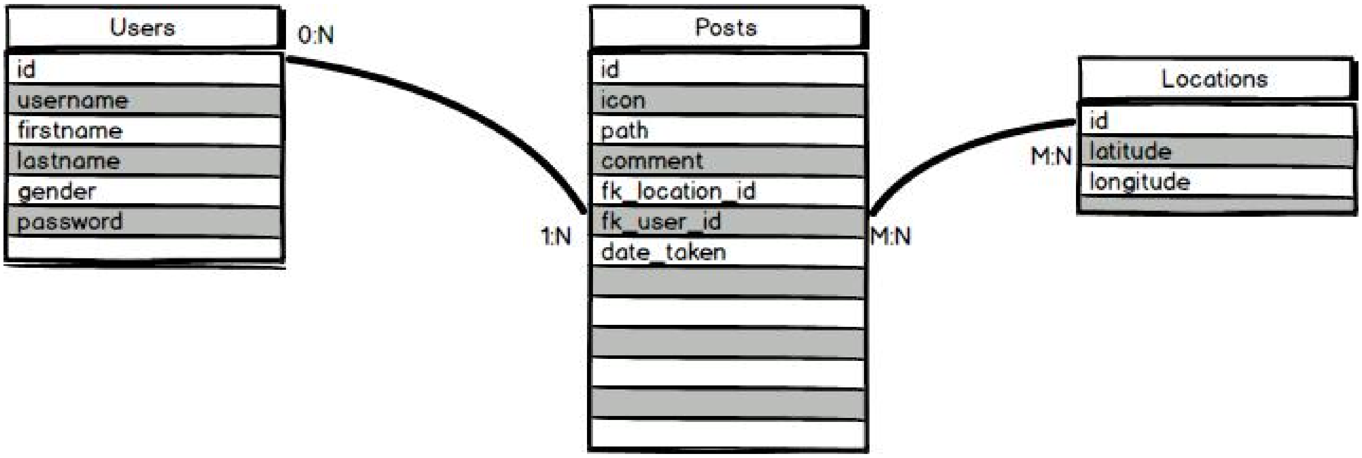


Figure 2: ER diagram.

Design Steps

Conceptualization

In this stage of the development, the team defined and refined the idea. We looked at four (4) factors which included:

1. the value it will bring to it's users,
2. the competitive advantage of the idea,
3. The integration with already built infrastructures
4. And how it will utilize I

These factors are crucial for the app to be successful. After these factors were realized we move on to the next stage.

UI/UX Prototype Design

Once the functionalities have been refined. A mock up of the user interface was designed according to the android design guidelines. The mock up was prepared in balsamiq and was used as a guideline throughout the development stage.

Database Design

The database was designed using an Entity Relational Diagram and implemented using a NoSQL with the ER diagram as a guideline, though the structure was implemented differently on the NoSQL platform. The development team choose this platform because of it's built-in real-time features and due to the fact that if structure of the schema needs to be changed, it could be changed without any hassle.

Development

During this stage of the project the application was coded using the Ionic 3 Framework which is built on top of the Angular 4 Javascript framework coupled with Cordova that enables us to interact with native functionalities of the device. The development team capitalized on the modular MVVM this framework offers as it enables a separation of logic while enabling us to better, test and manage our code. The team utilized the Ionic cloud platform to serve as a repository to manage our code. Every change made to the project was uploaded to the the cloud platform. The project was pushed to Github at the release of the version 1.0.

Testing

We chose a Iterative approach to development as after each feature of the App was developed we tested it to ensure it was functioning as intended. Features that did not use native phone functionalites were testing in the browser. However at the end of each page of the App tests where run on a few devices with different versions of android. Though Ionic provided a automated testing module, all test were manual done.

Deployment

The application was deployed to Github and can be cloned for testing purposes.