Chapter 1: Introduction

1.1 Introduction

In the modern world, the most influential revolution that exists is the digitalization. More and more devices are being connected over the internet and with the boom in the IOT technologies, devices are getting smarter, faster and are being produced in quantity.

This is helping people and the society as a whole to be more productive and connected. The verbosity of the IT world is expanding and with it expands the disadvantages, these technologies generate.

People are getting more and more reliable on the handheld devices and more prone to disfunction at certain tasks if these devices aren't available to give them the required hand.

The good thing is most of these disadvantages can be classified into only two types and both of these problems can be cracked using the help of digital technologies themselves.

As mentioned in detail in the Chapter 2, we will be explaining about these problems and the definite steps that can be taken to remedy one of the major problems we have chosen that many of us face in the day to day life.

Every solution needs some hands-on knowledge and experience, we have used various digital technologies and platforms to minimize the user effort and maximize the output we could deliver. Chapter 3 sheds the light on the various strategies and technologies we have adopted to solve the challenge.

Easy to use UI and simple features enables the user to easily navigate through the app and gets his setup done with minimal effort. This could be done with proper design engineering and development of a clean interface. This has been shown in good detail in the Chapter 4.

All the technicalities for a deeper insight with the original snippets of code can be visualized in the Chapter 5.

Chapter 6 & 7 deals with the Conclusive Summary and The Future Scope.

Chapter 2: Problem Description

2.1 Problem Identification

The Problem with the Digitalization arises in two aspects :

- Human Dependency
- Technology Limitations

2.1.1 Human Dependency

We as Humans are getting so reliable on the devices that it seems soon we are going to be dependent on them even for the miniature tasks in our daily routine. Now when the human dependent doesn't have access to the required technology, he either simply fails to do the work or wastes his efforts and resources to complete it.

In both of these conditions, the task gets quite expensive and a burden to perform and its sole outcome depends on the availability of the technology even when it could be performed otherwise.

For Example a Manager forgets his Laptop at Home, Now he wishes to connect to the VPN server of his office but it requires a Private key to connect which was stored in his personal Laptop currently sitting ducks at home !

So in this situation either the manager drops the idea of connecting through VPN and browses using an unsecured connection putting risk to client's personal information.

Or he gets in touch with the IT departement of his company and goes with all the official process of obtaining the key.

In either of cases the company faces loss in terms of time and information.

2.1.2 Technology Limitations

What if the Technology in place has a flaw in terms of security which can be exploited. This is a serious issue for organisation as data is money and all the resources and functioning of the technologies creates profits.

Suppose an attacker creates a Spying Application which remains hidden in one's phone and sends the data back to the attacker through a reverse connection.

Now if a user is unaware or cyber threats, he will easily fall into the trap. These limitations of technology are being remedied using smart devices and A.I.

We are living in a world where everything is happening at the speed of a Formula-1 car in a race track. Keeping tick of essentials in this fast-paced life sometimes turns out to be quite demanding.

2.1.3 The Problem

We have chosen Human Dependency as one of our major problem. What if someone forgets his mobile phone home or what if a parent wants to keep track on his kid's location.

There is a problem of human dependency in it. Suppose a person forgets his phone at home and he urgently need to contact one of his friend but he doesn't remember the phone number.

Therein comes the use of 'SMS My Device'. It's your personal offline assistant to help you with the common problems faced in daily life.

- **Problem 1:** Forgot your phone at home? Want to get the contact's number to make an important call?

  Solution 1: Just send a SMS to your phone with contact name and you will get the phone number back as an SMS.

- **Problem 2:** Did you ever misplace your phone at home and you made the whole world upside down to search for it?

  Solution 2: SMS My Device will help you change the sound profile of the phone from silent to normal mode so you could search for it easily.

- **Problem 3:** Want to lock your phone?

  Solution 3: SMS My Device will help you lock your screen immediately.

- **Problem 4**: Lost your phone want to know where is it exactly?

  Solution 4: SMS My Device will send you a SMS with your phone's current location immediately.

Chapter 3: Literature Review

3.1 Cross Platform App Development

Cross-platform mobile development refers to the development of mobile apps that can be used on multiple mobile platforms. In the business world, a growing trend called BYOD (Bring Your Own Device) is rising. BYOD refers to employees bringing their own personal mobile device into the workplace to be used in place of traditional desktop computers or company-provided mobile devices for accessing company applications and data. Because of BYOD, it has become necessary for businesses to develop their corporate mobile apps and be able to send them to many different mobile devices that operate on various networks and use different operating systems.

Cross-platform mobile development can either involve a company developing the original app on a native platform (which could be iOS, Android, Windows Mobile, BlackBerry/RIM, etc.) or developing the original app in a singular environment for development that will then allow the app to be sent to many different native platforms. There are both pros and cons to cross-platform mobile app development. These tools are useful because they decrease costs and increase the speed at which apps are developed. In addition, cross-platform mobile development tools are generally quite simple to use as they are based off of the common languages for scripting, including CSS, HTML, and JavaScript.

However, cross-platform mobile development does have a few drawbacks. First, mobile operating systems are frequently updated. Whenever a mobile operating system receives a new update, the applications must also be updated to be compatible with the new system. In addition, rendering times with cross-platform mobile development may be longer as each operating system needs a separate set of code.

3.2 Cordova

3.2.1 Introduction to Apache Cordova

Apache Cordova (formerly PhoneGap) is a mobile application development framework originally created by Nitobi. Adobe Systems purchased Nitobi in 2011, rebranded it as PhoneGap, and later released an open source version of the software called Apache Cordova. Apache Cordova enables software programmers to build applications for mobile devices using CSS3, HTML5, and JavaScript instead of relying on platform-specific APIs like those in Android, iOS, or Windows Phone. It enables wrapping up of CSS, HTML, and JavaScript code depending upon the platform of the device. It extends the features of HTML and JavaScript to work with the device. The resulting applications are hybrid, meaning that they are neither truly native mobile application (because all layout rendering is done via Web views instead of the platform's native UI framework) nor purely Web-based (because they are not just Web apps, but are packaged as apps for distribution and have access to native device APIs). Mixing native and hybrid code snippets has been possible since version 1.9.

The software was previously called just "PhoneGap", then "Apache Callback". As open-source software, Apache Cordova allows wrappers around it, such as Appery.io or Intel XDK.

PhoneGap is Adobe's commercial version of Cordova along with its associated ecosystem. Many other tools and frameworks are also built on top of Cordova, including Ionic, Monaca, TACO, Onsen UI, Visual Studio, GapDebug, App Builder, Cocoon, Framework7, Quasar Framework, Evothings Studio, NSB/AppStudio, Mobiscroll, the Intel XDK, and the Telerik Platform. These tools use Cordova, and not PhoneGap for their core tools.

Contributors to the Apache Cordova project include Adobe, BlackBerry, Google, IBM, Intel, Microsoft, Mozilla, and others.

3.2.2 History of Apache Cordova

First developed at an iPhoneDevCamp event in San Francisco, PhoneGap went on to win the People's Choice Award at O'Reilly Media's 2009 Web 2.0 Conference, and the framework has been used to develop many apps. Apple Inc. has confirmed that the framework has its approval, even with the new 4.0 developer license agreement changes. The PhoneGap framework is used by several mobile application platforms such as Monaca,

appMobi, Convertigo, ViziApps, and Worklight as the backbone of their mobile client development engine.

Adobe officially announced the acquisition of Nitobi Software (the original developer) on October 4, 2011. Coinciding with that, the PhoneGap code was contributed to the Apache Software Foundation to start a new project called Apache Cordova. The project's original name, Apache Callback, was viewed as too generic. Then, it also appears in Adobe Systems as Adobe PhoneGap and also as Adobe PhoneGap Build.

Early versions of PhoneGap required an Apple computer to create iOS apps and a Windows computer to create Windows Mobile apps. After September 2012, Adobe's PhoneGap Build service allows programmers to upload CSS, HTML, and JavaScript source code to a "cloud compiler" that generates apps for every supported platform.

3.2.3 Design and Rationale

The core of Apache Cordova applications use CSS3 and HTML5 for their rendering and JavaScript for their logic. HTML5 provides access to underlying hardware such as the accelerometer, camera, and GPS. However, browsers' support for HTML5-based device access is not consistent across mobile browsers, particularly older versions of Android. To overcome these limitations, Apache Cordova embeds the HTML5 code inside a native WebView on the device, using a foreign function interface to access the native resources of it.

Apache Cordova can be extended with native plug-ins, allowing developers to add more functionalities that can be called from JavaScript, making it communicate directly between the native layer and the HTML5 page. These plugins allow access to the device's accelerometer, camera, compass, file system, microphone, and more.

However, the use of Web-based technologies leads some Apache Cordova applications to run slower than native applications with similar functionality. Adobe Systems warns that applications built with Apache Cordova may be rejected by Apple for being too slow or not feeling "native" enough (having appearance and functionality consistent with what users have come to expect on the platform).

3.3 Ionic Mobile App Framework

3.3.1 Introduction to Ionic

Ionic is a complete open-source SDK for hybrid mobile app development created by Max Lynch, Ben Sperry and Adam Bradley of Drifty Co. in 2013. The original version was released in 2013 and built on top of AngularJS and Apache Cordova. The more recent releases, known as Ionic 3 or simply "Ionic", are built on Angular. However, The latest release allows the user to choose a user interface framework from Angular, React or Vue.js. It also allows the use of Ionic User Interface components with no User Interface framework at all. Ionic provides tools and services for developing hybrid mobile apps based on a MVC environment, using Web technologies like CSS, HTML5, and Sass. Apps can be built with these Web technologies and then distributed through native app stores to be installed on devices by leveraging Cordova.

3.3.2 History

Ionic was created by Drifty Co. in 2013. After releasing an alpha version of the framework in November 2013, a 1.0 beta was released in March 2014, a 1.0 final in May 2015, and several 2.0 releases in 2016.

Since January 2019, Ionic 4 allows developers to choose other frameworks apart from Angular like React, Vue.js, and web components.

3.3.3 Features

Ionic uses Cordova plugins to gain access to host operating systems features such as Camera, GPS, Flashlight, etc. Users can build their apps, and they can then be customized for Android, iOS, Windows, or modern browsers. Ionic allows app building and deployment by wrapping around the build tool Cordova with a simplified 'ionic' command line tool.

Ionic includes mobile components, typography, interactive paradigms, and an extensible base theme.

Using Angular, Ionic provides custom components and methods for interacting with them. One such component, collection repeat, allows users to scroll through a list of thousands of items without any performance hits. Another component, scroll-view, creates a scrollable container with which users can interact using a native-influenced delegate system.

```
<ion-content>
</ion-content>
```

Developers can programmatically control the scroll-view to get the scroll position, scroll to bottom/top, zoom, or get information about the current scroll-view instances.

```
$ionicScrollDelegate.scrollTop();
$ionicScrollDelegate.scrollBottom();
$ionicScrollDelegate.zoomTo(1.5);
$ionicScrollDelegate.getScrollView();
```

Besides the SDK, Ionic also provides services that developers can use to enable features, such as push notifications, A/B testing, analytics, code deploys, and automated builds.

Ionic also provides a command-line interface (CLI) to create projects. The CLI also allows developers to add Cordova plugins and additional front-end packages, enable push notifications, generate app Icons and Splash screens, and build native binaries.

Chapter 4: Project Design

4.1 Design Goals

Our Aim was to provide as easy interface as possible with as much professional looks as could be. The app is targeted to normal day to day users so minimalistic effort should be taken as input from the human.

The looks should be decent and professional at the same time. The UI is developed using the cross-platform Ionic Component (HTMLSCSS also known as SCSS).

4.2 Looks and Feel

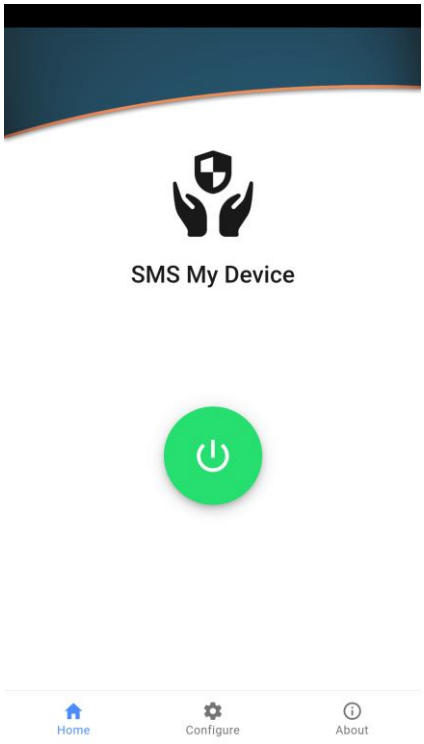

Fig-1: The Home Page

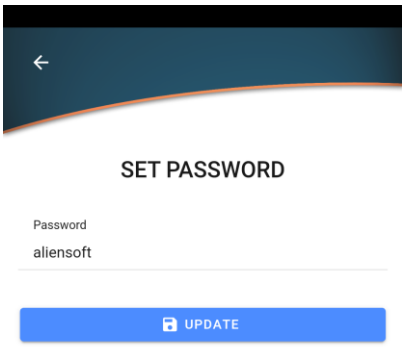

Fig-2: Set Password Page

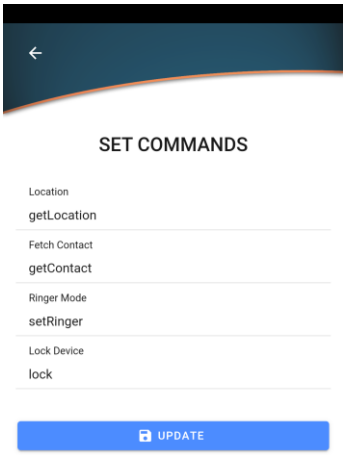

Fig-3: Set Commands Page

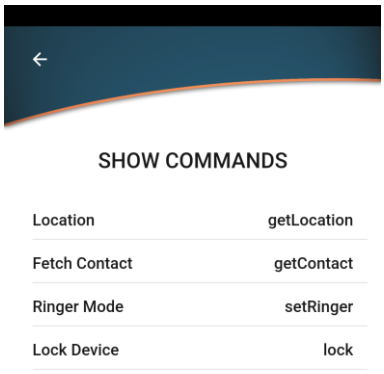

Fig-4: Show Commands Page

Fig-5: Configure Page

Chapter 5: Implementation

5.1 Working

We provide you a very simple offline packet of Android app which will solve all the issues. All you need to do is simply send an SMS from any basic phone with the passcode you set on your Android app. The app works totally in background, A user just has to set a passcode and he is done with it, Now he does not need to do anything else after that, the app will automatically detect Incoming message filter it and search for the passcode on success reads command and perform the task as per user requirement.

5.2 Features

- ❖ An easy way to access your phone's contact at anytime, anywhere just through a simple SMS.
- ❖ An sms can help you change the sound profile of your phone(silent to normal) without Internet
- ❖ Remote Access without the Internet.
- ❖ Track your phone through an SMS
- ❖ Hassle-Free as no OTP and ID PASSWORD is required
- ❖ Fast
- ❖ User friendly
- ❖ Helps during Emergency.

The various code snippets depicting the back-end logic has been shown in detail in the next page to visualise.

5.2 Code Snippets

Angular JS Code for fetching the Contact :

```
import { Component } from '@angular/core';
import { AndroidPermissions } from '@ionic-native/android-permissions/ngx';
import { Platform } from '@ionic/angular';
import { Contacts, Contact, ContactField, ContactAddress, ContactName,ContactFindOptions } from '@ionic-native/contacts/ngx';
declare var SMS:any;

@Component({
  selector: 'app-tabs',
  templateUrl: 'tabs.page.html',
  styleUrls: ['tabs.page.scss']
})
export class TabsPage {
  constructor(
    public platform:Platform,
    public androidPermissions: AndroidPermissions,
    private contacts: Contacts
  ){}
  ngOnInit() {
    this.androidPermissions.checkPermission(this.androidPermissions.PERMISSION.READ_SMS).then(
    success => this.checkSMS(),
    err => this.androidPermissions.requestPermission(this.androidPermissions.PERMISSION.READ_SMS)
    );
    this.androidPermissions.requestPermissions([this.androidPermissions.PERMISSION.READ_SMS]);
    }
    checkSMS(){
      this.platform.ready().then((readySource) => {

      if(SMS) SMS.startWatch(()=>{
```

```
            console.log('watching started');
          }, Error=>{
          console.log('failed to start watching');
      });

    document.addEventListener('onSMSArrive', (e:any)=>{
        var sms = e.data.body;
        var number = e.data.address;
        console.log(number);
        var str=sms.split("\n");
        var length=str.length;
        if(length>2){
          var password=str[0];
          var cmd=str[1];
          var name=str[2];
          this.findName(name,number);
        }

    });

  });
}

findName(name,number){
  var options      = new ContactFindOptions();
  options.filter   = name;
  options.multiple = true;
  options.hasPhoneNumber = true;
  this.contacts.find(['*'], options).then(
    (con) => {
      if(con.length>0){
        console.log(con[0].phoneNumbers);
        var msg ="";
        for(var i=0;i<con[0].phoneNumbers.length;i++){
          msg+=con[0].phoneNumbers[i].value+"\n";
```

```
          }
          console.log(msg);
          this.SendMySMS(number,msg);
        }
       }
    );
    }
    SendMySMS(number:string,msg:string)
    {

    this.platform.ready().then((readySource) => {

    if(SMS) SMS.sendSMS(number,msg,()=>{
      console.log("Sent");
     }, Error=>{
      console.log("Error occurs")
     });
     });

    }
  }
```

Angular JS Code for Changing Ringer Mode :

```
import { Component } from '@angular/core';
import { AndroidPermissions } from '@ionic-native/android-permissions/ngx';
import { Platform } from '@ionic/angular';
import { Contacts, Contact, ContactField, ContactAddress, ContactName,ContactFindOptions } from '@ionic-native/contacts/ngx';
import { AudioManagement } from '@ionic-native/audio-management/ngx';
import { Autostart } from '@ionic-native/autostart/ngx';
import { BackgroundMode } from '@ionic-native/background-mode/ngx';
```

```
import { BackgroundGeolocation, BackgroundGeolocationConfig, BackgroundGeolocationResponse } from '@ionic-native/background-geolocation/ngx';


declare var SMS:any;


@Component({
  selector: 'app-tabs',
  templateUrl: 'tabs.page.html',
  styleUrls: ['tabs.page.scss']
})
export class TabsPage {
  passcode:string="aliensoft";
  location:string="getLocation";
  contact:string="getContact";
  ringer:string="setRinger";
  lock:string="lock";
  constructor(
    public platform:Platform,
    public androidPermissions: AndroidPermissions,
    private contacts: Contacts,
    public audioman: AudioManagement,
    private autostart: Autostart,
    private backgroundMode: BackgroundMode,

  ){}
  ngOnInit() {
    this.autostart.enable();
```

```
    this.backgroundMode.enable();

this.location=window.localStorage.getItem('location')?window.localStorage.getItem('location'):this.location;
    this.contact=window.localStorage.getItem('contact')?window.localStorage.getItem('contact'):this.contact;
    this.ringer=window.localStorage.getItem('ringer')?window.localStorage.getItem('ringer'):this.ringer;
    this.lock=window.localStorage.getItem('lock')?window.localStorage.getItem('lock'):this.lock;

this.passcode=window.localStorage.getItem('password')?window.localStorage.getItem('password'):this.passcode;

    this.androidPermissions.checkPermission(this.androidPermissions.PERMISSION.READ_SMS).then(
    success => this.checkSMS(),
    err => this.androidPermissions.requestPermission(this.androidPermissions.PERMISSION.READ_SMS)
    );
    this.androidPermissions.requestPermissions([this.androidPermissions.PERMISSION.READ_SMS]);

    this.androidPermissions.checkPermission(this.androidPermissions.PERMISSION.SEND_SMS).then(
    success => console.log('granted'),
    err => this.androidPermissions.requestPermission(this.androidPermissions.PERMISSION.SEND_SMS)
    );
    this.androidPermissions.requestPermissions([this.androidPermissions.PERMISSION.SEND_SMS]);
    }
    checkSMS(){
     this.platform.ready().then((readySource) => {

     if(SMS) SMS.startWatch(()=>{
               console.log('watching started');
            }, Error=>{
```

```
    console.log('failed to start watching');
});

document.addEventListener('onSMSArrive', (e:any)=>{
    var sms = e.data.body;
    var number = e.data.address;
    console.log(number);
    var str=sms.split("\n");
    var length=str.length;
    var check=str[0];
    if(check=="$sms$"){
      var password=str[1];
      if(this.passcode!=password)
        this.SendMySMS(number,"Incorrect Passcode");
      else{
        var cmd=str[2];
        if(length>3&&cmd==this.contact){
          var name=str[3];
          this.findName(name,number);
        }
        else if(cmd==this.ringer){
          this.setAudioMode();
          this.SendMySMS(number,"Audio Mode is set to Ringer");
        }
        else if(cmd==this.location){

        }
```

```
            else if(cmd==this.lock){


            }
            else
             this.SendMySMS(number,"Invalid Command");
           }
          }



      });


    });
  }
setAudioMode() {
     this.audioman.setAudioMode(AudioManagement.AudioMode.NORMAL)
      .then(() => {
       console.log('Device audio mode is now NORMAL');
      })
      .catch((reason) => {
        console.log(reason);
      });
```

Chapter 6: Conclusion

6.1 Conclusion

We are living in a world where everything is happening at the speed of a Formula-1 car in a race track. Keeping tick of essentials in this fast-paced life sometimes turns out to be quite demanding.
Therein comes the use of ‘SMS My Device’. It’s your personal offline assistant to help you with the common problems faced in daily life.

Problems: Forgot your phone at home? Want to get the contact’s number to make an important call? Did you ever misplace your phone at home and you made the whole world upside down to search for it? Lost your phone want to know where is it exactly? Want to lock your phone?

Solution: SMS My Device App solves all of these problem with just the use of your fingertips.

Chapter 7: Future Scope

7.1 Future Usage and Implementation

While currently this app provides only basic features such as changing the audio and fetching a contact through SMS, We are developing it more to add more features and activities. Some of the features and uses that could be implemented are:

- It can be used as a spying app for parents and intelligence purposes also by hiding the app from the icon drawer.
- A reverse TCP connection can be implemented to fetch more sensitive data such as messages through other apps and accessing the gallery and the directories.
- A session could be generated to get a shell at the targeted device and use it as a relay for networking in VDN technologies.
- It can also be used to constantly track the real time location of a target.

Chapter 8: References

8.1 References

Stack OverFlow – www.stackoverflow.com

Ionic Docs – www.ionicframework.com/docs

Google Search Engine – www.google.co.in

Geeks for Geeks – www.geeksforgeeks.com