MPloyed

A

Minor Project Report

Submitted
In partial fulfilment
For the award of the Degree of

Bachelor of Technology (CSE-CT & IS)

In Department of Computer Science & Engineering

Submitted By:	Supervisor:	Submitted to:
Sanchit Bhatnagar (160101241)	Sushant Jhingran	Dr. Nitin Rakesh
Harshita Sachdeva (160101118)	Asst. Professor	HOD (CSE)

Department of Computer Science and Engineering
Sharda University, Greater Noida

January 2018

Candidate's Declaration

I hereby declare that the work, which is being presented in this report, entitled **"MPloyed"** in partial fulfilment for the award of Degree of "**Bachelor of Technology**" in department of Computer Science. **Sharda University** is a record of my own investigations carried under the Guidance of Mr Sushant Jhingran, Department of Computer Science Engineering.

I have not submitted the matter presented in this report anywhere for the award of any other Degree.

Sanchit Bhatnagar

Computer Science Engineering (CT & IS),
Enrollment No.: 2016005152

Harshita Sachdeva

Computer Science Engineering (CT & IS),
Enrollment No.: 2016004389

Counter Signed by
Sushant Jhingran

ACKNOWLEDGEMENT

We would like to pay our regards and heartfelt gratitude to Mr. Sushant Jhingran (Project Supervisor) and to my teachers specially Mr. Nitish Patil for giving right suggestions that makes the project fair and efficient. As my project guide, our teacher help me through their suggestions. We shall be highly obliged to them.

We also pay our regards to Prof. Dr. Nitin Rakesh (HOD computer science) for his ongoing support and encouragement in every aspect and for giving us platform to develop our career out there. We are also thankful to our parents for giving us financial support and appreciating us at every step.

The successful completion of our project would not have been possible without the dedicated support from all our mentors, family and friends.

Table of Contents

Table of Figures

ABSTRACT

In the current scenario, there is a rat race in each and every professional field. The same holds true for the job market. Our Job Portal is an application that has been sincerely dedicated so as to seek/compile/gather online information about recruiters as well as the job seekers.

Our job portal has been designed in such a manner that helps both the job seekers and the recruiters in finding the right organization for the employees. In the case of job seekers, according to their educational qualifications, experience, and their respective preferences - wherein the job seeker himself can find and apply for their targeted job. In addition, to the recruiters, provides the suitable candidate from a pool of lacks thereby creating a win-win for both the parties.

Therefore, our job portal would be a perfect online arena, where the job seekers and the employers find their goal in the pursuit of getting a top-notch company for the suitable candidates.

Chapter 1

1. INTRODUCTION

Our MPloyed application is a Multi-Platform mobile application which allows applicants and employers to register their own respective details. Applicants can browse through the vacancy details that are posted and can apply for the suited jobs online according to the area selected. It includes a filter search facility for the job seekers according to their required Locations.

Sending resumes saves effort, time and cost of the job seeker. All the vacancies are available on a single interface job seeker thereby easing out the communication between the job seeker and the employer.

Chapter 2

2. REQUIREMENT ANALYSIS (SOFTWARE & HARDWARE)

2.1 HARDWARE REQUIREMENTS:

2.1.1 Android Device:
- 4.1.1.1 RAM: 2 GB or higher
- 4.1.1.2 Android Version : 6.0 or higher

2.1.2 IOS Device:
- 4.1.2.1 IOS Version: 10.0.2 or higher

2.1.3 Computer Device:
- 4.1.2.1 RAM: Minimum – 4Gb Recommended – 6Gb
- 4.1.2.2 Graphic memory - 2Gb recommended
- 4.1.2.3 Windows 7 or higher Or Ubuntu or Mac OS
- 4.1.2.4 Storage Requirements – 10Gb
- 4.1.2.5 Processor- Any processor with minimum 1 GHz processing speed.

2.2 SOFTWARE REQUIREMENTS:

2.2.1 React Native CLI
2.2.2 Expo XDE
2.2.3 Node.js
2.2.4 Text Editor

2.3 PREREQUISITES:

3.2.1 Basic JavaScript Knowledge
3.2.2 Advanced ReactJS Knowledge
3.2.3 React Native Knowledge
3.2.4 Knowledge for Applications in Android and Android Studio
3.2.5 Knowledge of Applications in IOS Devices

Chapter 3

3. DESIGN DOCUMENT

3.1 Background

What is React Native?
React Native is a framework developed by Facebook for creating native-style apps for iOS & Android under one common language, JavaScript. Initially, Facebook only developed React Native to support iOS. However with its recent support of the Android operating system, the library can now render mobile UIs for both platforms.

Why React Native ?

Whenever there is an update for apps written in Swift/Objective-C or Java, the whole app needs to be recompiled and a new version has to be distributed to the App Store again. All this can take a few weeks depending on the App Store review process.

3.2 Modules

3.2.1 Welcome Screens Module :

This set of screens will be swipe able and
Will give an overview of what the app is for

3.2.2 Facebook Auth Screen :

This screen will perform Facebook
Authentication to login user to the app

3.2.3 Map Screen:

This screen will show a map from which
The user can choose and search list of jobs
From chosen location

3.2.4 Jobs List Screen:

This screen fetches list from API and
Shows jobs from the chosen location.
Jobs are saved by swiping right and
Ignored by swiping left.

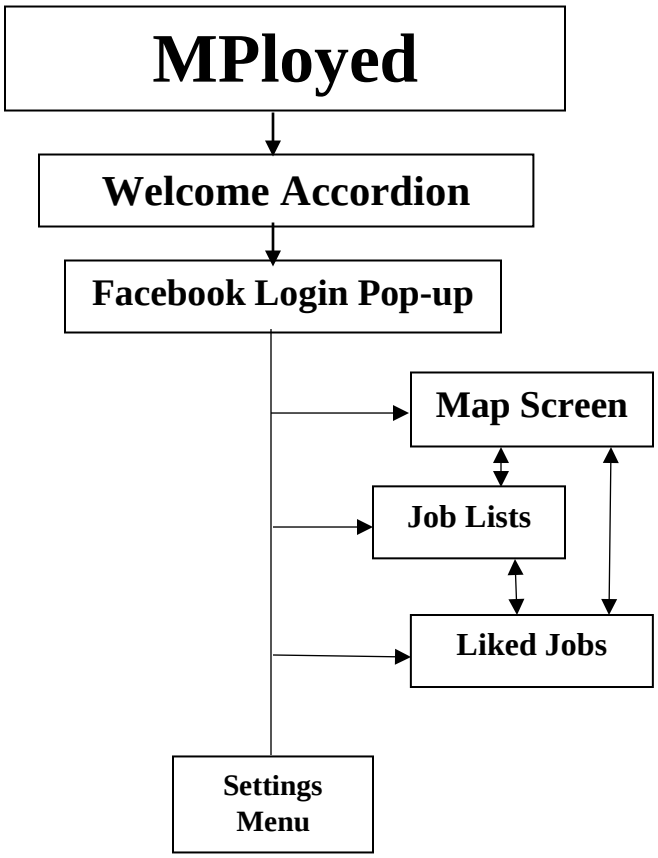
3.2.5 Review Screen:

This screen shows saved jobs and users
Can go to company's job portal to apply
To their jobs

3.2.6 Settings Screen :

This screen has 2 buttons:
Logout: Logs out user and returns to Welcome screen

3.3 FLOWCHART



Chapter 4

4. CODING IMPLEMENTATION

4.1 Code :

4.1.1 ROOT CONFIG FILE (app.json)

```
{
          "expo": {
            "name": "MPloyed",
            "description": "Provides Job to People Location Wise",
            "slug": "mployed-app",
            "privacy": "unlisted",
            "sdkVersion": "29.0.0",
            "platforms": ["ios", "android"],
            "version": "1.2.0",
            "orientation": "portrait",
            "icon": "./assets/ic_mployed.png",
            "githubUrl":"https://github.com/SanchitB23/MPloyed",
            "splash": {
              "image": "./assets/jobSplash.png",
              "resizeMode": "cover",
              "backgroundColor": "#fff"
            },
            "updates": {
              "fallbackToCacheTimeout": 0
            },
            "assetBundlePatterns": [
              "**/*"
            ],
            "ios": {
              "supportsTablet": true,
              "bundleIdentifier": "com.mrawesome.mployed"
            },
            "android": {
              "package": "com.mrawesome.mployed",
              "versionCode":2,
              "config": {
                "googleMaps": {
                  "apiKey": "AIzaSyCAVe1LuyNODmnWZgLKu9an4F6fXnLaiBQ"
                }
              }
            }
          }
```

4.1.2 MAIN LAUNCHER FILE (app.js)

```
import React from 'react';
import { createBottomTabNavigator, createStackNavigator } from 'react-
navigation';
import { Provider } from 'react-redux';
import { Icon } from 'react-native-elements';

import store from './src/store';
import AuthScreen from './src/screens/AuthScreen';
import WelcomeScreen from './src/screens/WelcomeScreen';
import MapScreen from './src/screens/MapScreen';
import DeckScreen from './src/screens/DeckScreen';
import SettingScreen from './src/screens/SettingScreen';
import ReviewScreen from './src/screens/ReviewScreen';

export default class App extends React.Component {
  render() {
    const MainNavigator = createBottomTabNavigator({
      welcome: { screen: WelcomeScreen },
      auth: { screen: AuthScreen },
      main: createBottomTabNavigator({
          map: { screen: MapScreen },
          deck: { screen: DeckScreen },
          review: {
            screen: createStackNavigator({
                review: {
                  screen: ReviewScreen,
                },
                settings: { screen: SettingScreen }
              }),
              navigationOptions: {
                tabBarLabel: 'Review Jobs',
                tabBarIcon: ({ tintColor }) => (
                  <Icon name="favorite" size={30} color={tintColor} />
                ),              }
            }
      }, {
        tabBarOptions: {
          labelStyle: { fontSize: 12 }
        }
      })
    }, {
      navigationOptions: {
        tabBarVisible: false
      },
      lazyLoad: true //not preload Components in navi
    });
    return (
      <Provider store={store}>
        <MainNavigator />
      </Provider>
    );
  }
}
```

4.1.3 FACEBOOK AUTH FILE (AuthScreen.js)

```
import React, { Component } from 'react';
import { View } from 'react-native';
import { connect } from 'react-redux';

import * as actions from '../actions';

class AuthScreen extends Component {
  componentDidMount() {
    this.props.facebookLogin();
    this.onAuthComplete(this.props);
  }

  componentWillReceiveProps(nextProps) {
    this.onAuthComplete(nextProps);
  }
  onAuthComplete(props) {
    if (props.token) {
      this.props.navigation.navigate('map');
    }
  }
  render() {
    console.log('auth');
    return (
      <View />
    );  }
}
function mapStateToProps({ auth }) {
  return { token: auth.token };
}
export default connect(mapStateToProps, actions)(AuthScreen);
```

4.1.4 MAP SCREEN FILE (MapScreen.js)

```
import React, { Component } from 'react';
import { View, ActivityIndicator } from 'react-native';
import { MapView } from 'expo';
import { connect } from 'react-redux';
import { Button, Icon } from 'react-native-elements';
import * as actions from '../actions';

class MapScreen extends Component {
  static navigationOptions={
    title: 'Map',
    tabBarIcon: ({ tintColor }) => (
      <Icon name="my-location" size={30} color={tintColor} />    )}
  state={
```

```
    mapLoaded: false,
    region: {
      longitude: 77.216721,
      latitude: 28.644800,
      longitudeDelta: 0.04,
      latitudeDelta: 0.09
    },
    loading: false}

  componentDidMount() {
    this.setState({ mapLoaded: true });  }
  onRegionChangeComplete=(region) => {
    this.setState({ region });  }
  onButtonPress=async() => {
    try {
    this.setState({ loading: true });
      await this.props.fetchJobs(this.state.region, () => {
        this.props.navigation.navigate('deck', {
          lat: this.state.region.latitude,
          long: this.state.region.longitude        });      });
      this.setState({ loading: false });
    } catch (e) {      console.log(e);    }
  }
  render() {
    if (!this.state.mapLoaded) {
      return (
        <View style={{ flex: 1, justifyContent: 'center' }}>
          <ActivityIndicator            size="large"          />
        </View>
      );
    }
    return (
      <View style={{ flex: 1 }}>
        <MapView
          style={{ flex: 1 }}
          region={this.state.region}
          onRegionChangeComplete={this.onRegionChangeComplete}
        />
        <View style={styles.buttonContainer}>
          <Button
            large
            backgroundColor='#009688'
            title="Search This Area"
            loading={this.state.loading}
            icon={{ name: 'search' }}
            onPress={this.onButtonPress}           />
        </View>
      </View>    );  }}

export default connect(null, actions)(MapScreen);

const styles = {
  buttonContainer: {
    position: 'absolute',
    bottom: 20,
    left: 0,
    right: 0  }};
```

4.1.5 JOB LIST SCREEN FILE (DeckScreen.js)

```
import React, { Component } from 'react';
import { View, Text, Platform } from 'react-native';
import { connect } from 'react-redux';
import { MapView } from 'expo';
import { Card, Button, Icon } from 'react-native-elements';
import * as actions from '../actions';
import Swipe from '../components/Swipe';

let latitude = '';
let longitude = '';
class DeckScreen extends Component {
  static navigationOptions={
    title: 'Jobs',
    tabBarIcon: ({ tintColor }) => (
      <Icon name="description" size={30} color={tintColor} />
    )
  }
  renderCard(job) {
    const initialRegion = {
      longitude,
      latitude,
      latitudeDelta: 0.045,
      longitudeDelta: 0.02
    };
    return (
      <Card
        title={job.title}
      >
        <View style={{ height: 300 }}>
          <MapView
            scrollEnabled={false}
            style={{ flex: 1 }}
            cacheEnabled={Platform.OS === 'android'}
            initialRegion={initialRegion}
          />
        </View>
        <View style={styles.detailWrapper}>
          <Text>{job.company}</Text>
          <Text>{job.created_at}</Text>
        </View>
        <Text>{job.description}</Text>
      </Card>
    );
  }
  renderNoMoreCards=() => (
      <Card
        title="No More Jobs"
      >
        <Button
          title='Back to Map'
```

```
          large
          icon={{ name: 'my-location' }}
          onPress={() => this.props.navigation.navigate('map')}
        />
      </Card>
    )
  render() {
    latitude = this.props.navigation.getParam('lat', 28.632744);
    longitude = this.props.navigation.getParam('long', 77.219597);
    console.log('DeckScreen');
    return (
      <View style={{ marginTop: 10 }}>
        <Swipe
          data={this.props.jobs}
          renderCard={this.renderCard}
          renderNoMoreCards={this.renderNoMoreCards}
          onSwipeRight={job => this.props.likeJob(job)}
        />
      </View>
    );
  }
}

function mapStateToProps({ jobs }) {
  return { jobs: jobs.data };
}

export default connect(mapStateToProps, actions)(DeckScreen);
const styles = {
  detailWrapper: {
    flexDirection: 'row',
    justifyContent: 'space-around',
    marginVertical: 10
  }
};
```

4.1.6 REVIEW JOBS SCREEN FILE (ReviewScreen.js)

```
import React, { Component } from 'react';
import { View, Text, Linking, Image, ScrollView } from 'react-native';
import { Button, Card } from 'react-native-elements';
import { connect } from 'react-redux';
import * as actions from '../actions';

class ReviewScreen extends Component {
  static navigationOptions =({ navigation }) => ({
```

```
 headerTitle: 'Review Jobs',
    headerRight: (
      <Button
        title='Settings'
        tabBarLabel='Settings'
        onPress={() => navigation.navigate('settings')}
        color="rgba(0,122,255,1)"
        backgroundColor="rgba(0,0,0,0)"
      />
    ),    })
  renderLikedJobs() {
    return this.props.likedJob.map((job, i) => (
        <Card key={i} title={job.title}>
          <View style={{ height: 200 }}>
            <Image
              source={{ uri: job.company_logo }}
              resizeMode='center'
              style={{ flex: 1, justifyContent: 'center', alignItems:
'center', height: 50 }}
            />
            <View style={styles.detailWrapper}>
              <Text style={styles.italics}>{job.company}</Text>
              <Text style={styles.italics}>{job.created_at}</Text>
            </View>
            <Button
              title='Apply Now'
              backgroundColor='#03A9F4'
              onPress={() => Linking.openURL(job.url)}
            />
          </View>
        </Card>
      )    );  }
  render() {
    return (
      <ScrollView>
        {this.renderLikedJobs()}
      </ScrollView>
    );  }}

function mapStateToProps({ likedJob }) {
  return { likedJob };
}
export default connect(mapStateToProps, actions)(ReviewScreen);

const styles = {
  detailWrapper: {
    marginVertical: 10,
    flexDirection: 'row',
    justifyContent: 'space-around',
  },
  italics: {
    fontStyle: 'italic'
  }
};
```

4.1.7 SETTINGS SCREEN FILE (SettingScreen.js)

```
import React, { Component } from 'react';
import { View, AsyncStorage, Linking } from 'react-native';
import { connect } from 'react-redux';
import { Button, Card } from 'react-native-elements';
import { clearLikedJobs } from '../actions';

const FEEDBACK_URL =
'https://docs.google.com/forms/d/e/1FAIpQLSdiUw6GIIU_RDz0vWM2bxe6dYomSeKDJmuT
ZAtpkiE_iaOwgg/viewform?usp=sf_link';

class SettingScreen extends Component {
  render() {
    return (
      <View>
        <Card>

          <Button
            title='Reset Liked Jobs'
            large
            icon={{ name: 'delete-forever' }}
            backgroundColor='#F44336'
            onPress={this.props.clearLikedJobs}
          />
        </Card>
        <Card>

          <Button
            title='Logout'
            large
            icon={{ name: 'delete-forever' }}
            backgroundColor='#F44336'
            onPress={() => {
              AsyncStorage.removeItem('fb_token');
              this.props.clearLikedJobs();
              this.props.navigation.navigate('welcome');
            }}
          />
        </Card>
        <Card>

          <Button
            title='Feedback'
            large
            icon={{ name: 'feedback' }}
            backgroundColor='#F9A825'
            onPress={() => Linking.openURL(FEEDBACK_URL)}
          />
        </Card>
      </View>
    );  }}

export default connect(null, { clearLikedJobs })(SettingScreen);
```

4.1.8 BACK END JOBS FETCH FROM API FILE (job_actions.js)

```
import axios from 'axios';
import {
  FETCH_JOBS,
  LIKE_JOB,
  CLEAR_LIKED_JOB
} from '../constants';
const URL = 'https://jobs.github.com/positions.json?markdown=true';
export const fetchJobs = (region, callbackForNav) => async (dispatch) => {
  try {
    const { latitude, longitude } = region;
    const url = `${URL}&lat=${latitude}&long=${longitude}`;
    const data = await axios.get(url);
    dispatch({ type: FETCH_JOBS, payload: data });
    callbackForNav();}
catch (e) {    console.error(e);  }};
export const likeJob = (job) => ({    type: LIKE_JOB,    payload: job  });

export const clearLikedJobs = () => ({ type: CLEAR_LIKED_JOB });
```

4.1.9 BACK END FACEBOOK AUTH CHECK FILE (auth_actions.js)

```
import { AsyncStorage } from 'react-native';
import { Facebook } from 'expo';
import {
  FACEBOOK_LOGIN_SUCCESS,
  FACEBOOK_LOGIN_FAIL
} from '../constants';

export const facebookLogin = () => async dispatch => {
  const token = await AsyncStorage.getItem('fb_token');

  if (token) {
    dispatch({ type: FACEBOOK_LOGIN_SUCCESS, payload: token });
  } else {
    doFacebookLogin(dispatch);
  }};

const doFacebookLogin = async dispatch => {
  const { type, token } = await
Facebook.logInWithReadPermissionsAsync('205415063460000', {
    permissions: ['public_profile']
  });

  if (type === 'cancel') {
    return dispatch({ type: FACEBOOK_LOGIN_FAIL });
  }
  await AsyncStorage.setItem('fb_token', token);
  dispatch({ type: FACEBOOK_LOGIN_SUCCESS, payload: token });};
```

4.1.10 BACK END LINKING AND METADATA FILE (store.js)

```
import { createStore, compose, applyMiddleware } from 'redux';
import thunk from 'redux-thunk';
import { persistStore, autoRehydrate } from 'redux-persist';
import { AsyncStorage } from 'react-native';

import reducers from '../reducers';

const store = createStore(
  reducers,
  {},
  compose(
    applyMiddleware(thunk),
    autoRehydrate()
  )
);

// persistStore(store, { storage: AsyncStorage, whitelist: ['likedJob']
}).purge();
// REVIEW: for production

persistStore(store, { storage: AsyncStorage, whitelist: ['likedJob'] });
export default store;
```

4.2 ScreenShot Library :

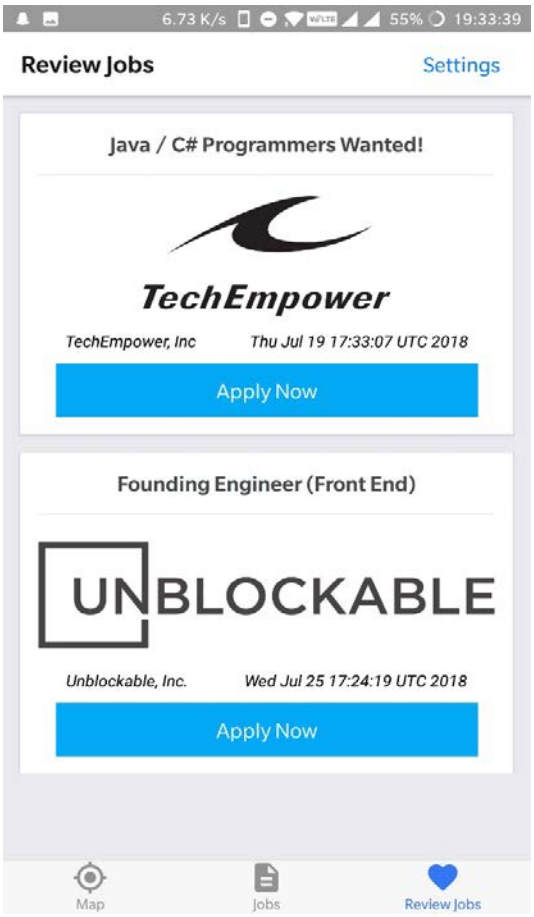


Image 4.2.1

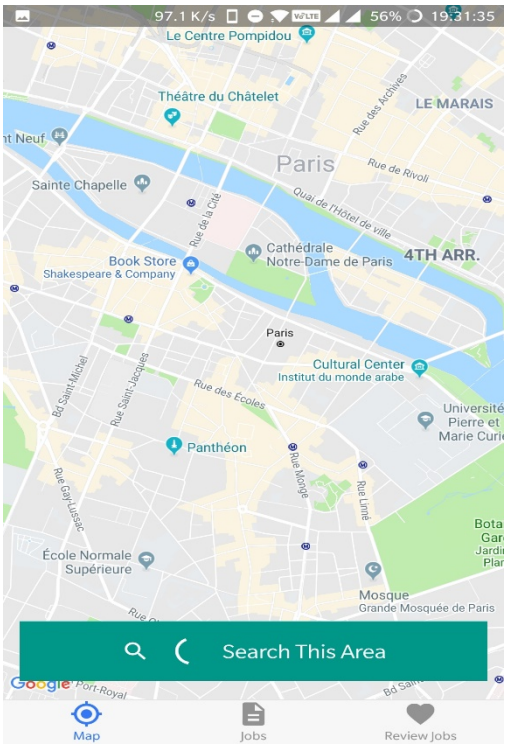


Image 4.2.2

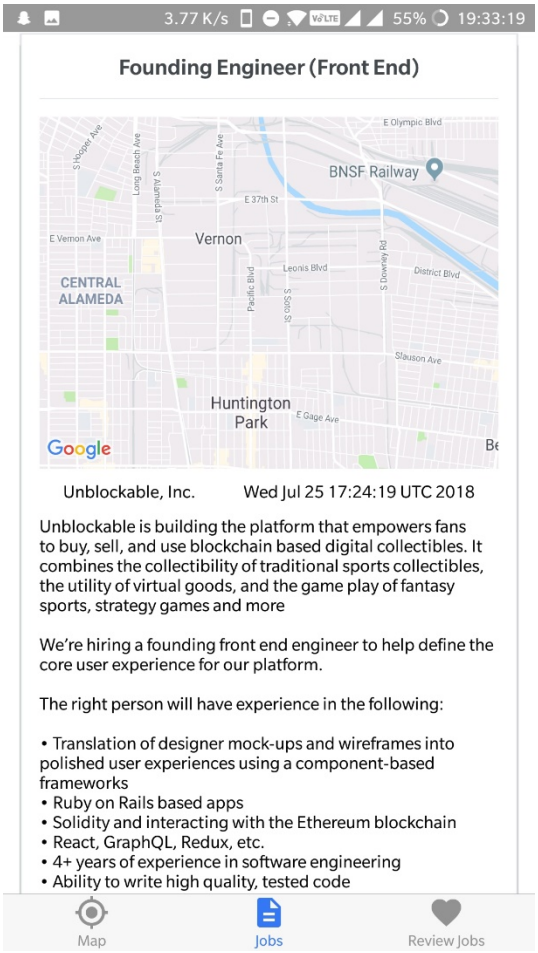

3.77 K/s
55%
19:33:19
Founding Engineer (Front End)
E Olympic Blvd
BNSF Railway
E 37th St
E Vernon Ave
Vernon
CENTRAL ALAMEDA
Leonis Blvd
District Blvd
Slauson Ave
Huntington Park
E Gage Ave
Google
Unblockable, Inc.
Wed Jul 25 17:24:19 UTC 2018
Unblockable is building the platform that empowers fans to buy, sell, and use blockchain based digital collectibles. It combines the collectibility of traditional sports collectibles, the utility of virtual goods, and the game play of fantasy sports, strategy games and more
We're hiring a founding front end engineer to help define the core user experience for our platform.
The right person will have experience in the following:
• Translation of designer mock-ups and wireframes into polished user experiences using a component-based frameworks
• Ruby on Rails based apps
• Solidity and interacting with the Ethereum blockchain
• React, GraphQL, Redux, etc.
• 4+ years of experience in software engineering
• Ability to write high quality, tested code
Map
Jobs
Review Jobs

Image 4.2.3

Chapter 5

5. CONCLUSION

An easy maintenance of records of various Recruiters (Companies), job and job seekers has been maintained. To check for the detailed prospective, the jobseekers can do so through quick search provided in the portal. An attempt to prevent and reduce human error.
Reduced manual work. Classic functionality focuses on data storage. In addition, it also outlines the means to retrieve and analyse data to extract, transform, load and process data.

Chapter 6

6. FUTURE SCOPE

The project has a very vast scope in future. There still have been few noted cases of device glitches, errors in content, and security lapses reported in online job searching. So, in the near future, we could foresee and make the application more secure and reliable. While electronic glitches are rare, they have been known to occur, for instance when computer crashes while voiding the efforts of numerous employees and companies. There have also been cases wherein corrupted databases have been reported. And therefore, our software can be programmed well so as to reduce the possibility of any such scenario to zero.

Alongside, the following can also be taken into consideration while keeping in mind the future scope for the same :

- Maintain Job Seeker and Employer records.
- Maintain the uploaded Resumes.
- Provide customized job postings.
- Maintain job posting details and generate various reports on regular basis.

It will also contribute to the reduced costing and collecting the management and collection procedure will thereby go on smoothly. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. With the proposed software of database ready and fully functional the client will be able to manage and hence run the entire work in a much better, accurate and error free manner.

7. REFERENCES

About the App:

[1] GitHub Repository – https://github.com/SanchitB23/MPloyed
[2] App Download Link - https://www.amazon.com/SanchitB23-MPloyed/dp/B07GD4RMZF/ref=sr_1_1?ie=UTF8&qid=1543216077&sr=8-1&keywords=mployed

Toolkits:

[1] Node Package Manager - https://nodejs.org/en/
[2] Expo XDE - https://docs.expo.io/versions/latest/introduction/installation
[3] React Native CLI - https://facebook.github.io/react-native/docs/getting-started.html
[4] Sublime Text Editor - https://www.sublimetext.com/3

Websites for Reference:

[1] Expo XDE Docs - https://docs.expo.io/
[2] React Native - https://facebook.github.io/react-native/