TAKEMETO HOSPITAL

INFO 6350 - Smartphone Based Web Development

TARUN KESWANI – NUID #001685431

MOHIT MITTAL – NUID #001617966

INTRODUCTION

TAKEMETO HOSPITAL is an app which provides user with all the hospital location on the map within 2 mile radius from his current location. The purpose of this app is to help user find hospitals nearby in the time of emergency. It will provide user with the distance and the wait time in hospital which will help user to take the decision of selecting the hospital he wants to visit. Also his past routes saved on the server will be shown to him which will help him in selecting the hospital. The app tends to help users provide important information that will help him in the case of emergency.

FEATURES

1. Display all the hospitals within the 2 mile radius of user correct location.
2. Showing distance and estimated time of travel from the user's current location on annotation to the selected hospital.
3. Selecting the hospital and providing different routes to user to reach the destination.
4. Display past time travel and waiting time of the hospital.

API USED –

1. **Facebook API -** We have used Facebook API to login assuming every user has a facebook account. Using the facebook API we get the user id and store it in the database to store the routes of the user. This user id becomes the differentiating factor when picking up the user past data.
2. **Factual Framework –** As part of the app, we showed all the hospitals nearby to user current location. We have used factual API to get the hospital data from the framework which we have plotted on the map. We have given annotations on each point which tells the distance of the hospital from user current location and the ETA (expected time of arrival). We get each hospital as an object which we show on the map.
3. **MapKit –** We have used Apple map to show the hospital location and default routes suggested by the map. We have embedded the maps into our views so that it doesn't have to open another application to display the map.

FACEBOOK API CODE

```
- (void)viewDidLoad {
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically from a nib.
    FBSDKLoginButton *loginButton = [[FBSDKLoginButton alloc] init];
    loginButton.center = self.view.center;
    loginButton.readPermissions = @[@"public_profile", @"email", @"user_friends"];
    loginButton.delegate = self;
    [self.view addSubview:loginButton];
}

-(void) viewDidAppear:(BOOL)animated{
    if ([FBSDKAccessToken currentAccessToken] != nil)
    {
        NSLog(@"User Logged in");
        [self getFacebookProfileInfos];
        [self goToNewViewController];
    }

}

-(void)getFacebookProfileInfos {

    FBSDKGraphRequest *requestMe = [[FBSDKGraphRequest alloc]initWithGraphPath:@"me" parameters:nil];

    FBSDKGraphRequestConnection *connection = [[FBSDKGraphRequestConnection alloc] init];
    [connection addRequest:requestMe completionHandler:^(FBSDKGraphRequestConnection *connection, id result, NSError *error) {

        if(result)
        {
            if ([result objectForKey:@"email"]) {
                NSLog(@"Email: %@",[result objectForKey:@"email"]);
            }
            if ([result objectForKey:@"name"]) {
                NSLog(@"Name : %@",[result objectForKey:@"name"]);
            }
            if ([result objectForKey:@"id"]) {
                NSLog(@"User id : %@",[result objectForKey:@"id"]);
            }

        }

    }];
    [connection start];
}
```

FACTUAL CODE

```
//creating query for API request
-(void) doQuery{
    NSString *cityName = @"Boston";
    if(cityName.length !=0){


        queryObject = [FactualQuery query];

        //change to current location of user
        CLLocationCoordinate2D coordinate = {mapLocation.coordinate.latitude, mapLocation.coordinate.longitude};
        [queryObject setGeoFilter:coordinate radiusInMeters:3200];
        [queryObject addRowFilter:[FactualRowFilter fieldName:@"category_ids" equalTo:@"74"]];

        queryObject.offset =offset;
        queryObject.limit = 50;
        apiObject = [AppDelegate getAPIObject];
        activeRequest = [apiObject queryTable:@"places-us" optionalQueryParams:queryObject withDelegate:self];
    }

}
```

RESULT FROM FACTUAL API

```
//receiving results for query from the API
-(void) requestComplete:(FactualAPIRequest *)request receivedQueryResult:(FactualQueryResult *)queryResultObj {
    if (activeRequest == request && noMoreResults == NO) {
        NSLog(@"Active request Completed with row count:%ld TableId:%@ RequestTableId:%@", (long)[queryResultObj rowCount], queryResultObj.tableId,activeRequest.
            tableId);


        self.queryResult = queryResultObj;


        activeRequest = nil;

        long numOfResults = [queryResult.rows count];
        if ([queryResult.rows count] > 0){
            int i;
            self.navigationItem.title = @"Boston";
            for (i = 0; i < numOfResults; i++) {
                FactualRow* row = [self.queryResult.rows objectAtIndex:i];
                Hospital *h = [[Hospital alloc]init];
                h.hospitalName =[row valueForName:@"name"];
                h.address = [NSString stringWithString:[row valueForName:@"address"]] ;
                 h.latitude = [row valueForName:@"latitude"];
                h.longitude = [row valueForName:@"longitude"];
                h.location = [[CLLocation alloc]initWithLatitude:[h.latitude doubleValue] longitude:[h.longitude doubleValue]];
                [hospitalList.hospitalList addObject:h];
                [self mapView:h];
            }

        }else{
            if([hospitalList.hospitalList count] == 0){
                UIAlertView *alert = [[UIAlertView alloc]initWithTitle:@"No Results Found! Please Check Your internet connection" message:@"" delegate:self
                    cancelButtonTitle:@"OK" otherButtonTitles:nil];
                [alert show];
            }else{
                UIAlertView *alert = [[UIAlertView alloc]initWithTitle:@"No More Results" message:@"" delegate:self cancelButtonTitle:@"OK" otherButtonTitles:nil];
                [alert show];
                noMoreResults = YES;
            }

        }

    }
}
```

RENDERING CODE

```
-(void) initialMapView{
    //called from view did load
    //Add Marker
    MyMKPointAnnotation *annotation = [[MyMKPointAnnotation alloc] init];
    [annotation setCoordinate:selectedHospital.location.coordinate];
    annotation.title=selectedHospital.hospitalName;
    annotation.subtitle=selectedHospital.address;
    annotation.hospital = selectedHospital;
    [routeMapView addAnnotation:annotation];



    MKPlacemark *sourcePlaceMark = [[MKPlacemark alloc] initWithCoordinate:location.coordinate addressDictionary:nil];
    MKMapItem *sourceMapItem = [[MKMapItem alloc] initWithPlacemark:sourcePlaceMark];


    MKDirectionsRequest *request1 = [[MKDirectionsRequest alloc] init];
//    [request1 setSource:[MKMapItem mapItemForCurrentLocation]];
    [request1 setSource:sourceMapItem];
    MKPlacemark *hospitalPlacemark = [[MKPlacemark alloc] initWithCoordinate:selectedHospital.location.coordinate
        addressDictionary:nil];

    MKMapItem *hospitalMapItem = [[MKMapItem alloc] initWithPlacemark:hospitalPlacemark];
    [request1 setDestination:hospitalMapItem];
    [request1 setTransportType:MKDirectionsTransportTypeAny]; // This can be limited to automobile and walking directions.
    [request1 setRequestsAlternateRoutes:NO]; // Gives you several route options.
    MKDirections *directions = [[MKDirections alloc] initWithRequest:request1];


        [directions calculateDirectionsWithCompletionHandler:^(MKDirectionsResponse *response, NSError *error) {
//            NSLog(@"Error Description: %@", error.description);
            if (!error) {
                for (MKRoute *route in [response routes]) {
                    [routeMapView addOverlay:[route polyline] level:MKOverlayLevelAboveRoads];
                }
```

INSTALLATION OF SERVER COMPONENT

The Spring Tool Suite is an Eclipse-based development environment that is customized for developing Spring applications. It provides a ready-to-use environment to implement, debug, run, and deploy your Spring applications, including integrations for Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ, and comes on top of the latest Eclipse releases.

Included with the Spring Tool Suite is the developer edition of Pivotal tc Server, the drop-in replacement for Apache Tomcat that's optimized for Spring. With its Spring Insight console, tc Server Developer Edition provides a graphical real-time view of application performance metrics that lets developers identify and diagnose problems from their desktops.

The Spring Tool suite supports application targeting to local, virtual and cloud-based servers. It is freely available for development and internal business operations use with no time limits, fully open-source and licensed under the terms of the Eclipse Public License.

PIVOTAL TC SERVER

Pivotal and VMware are partnering to deliver well-managed, agile and secure data and application infrastructure for VMware environments across public, private and hybrid clouds. Pivotal tc Server formerly known as VMware vFabric tc Server, is now part of the Pivotal portfolio of products.

Pivotal tc Server provides enterprise users with a lightweight Java application server that extends Apache Tomcat for use in large-scale mission-critical environments.

EXECUTE PIVOTAL SERVER

Running the server side code on Spring Tool Suite (STS) which have pivotal server.

Steps to run the pivotal server –

1. Drop down the Servers package in STS.
2. Select the Pivotal Server.
3. Right click Pivotal Server and select “Run As”.
4. This will start the server and server component is ready to be exceuted.

REQUEST TO SERVER TO GET PAST ROUTES

```
-(NSMutableArray *)getUserRouteFromServer:(NSNumber *) uid{
    NSMutableArray *userRouteListFromServer = [[NSMutableArray alloc]init];

    NSURL *url = [NSURL URLWithString:@"http://192.168.43.67:8565/myapp/suggestion"];
    NSMutableURLRequest *suggestRouteRequest = [NSMutableURLRequest requestWithURL:url];
    [suggestRouteRequest setHTTPMethod:@"POST"];
    [suggestRouteRequest setValue:@"application/json" forHTTPHeaderField:@"Accept"];
    NSString *srclat, *srclong, *destlat, *destlong, *fbid;
    srclat = [[NSNumber numberWithDouble:location.coordinate.latitude] stringValue];
    srclong = [[NSNumber numberWithDouble:location.coordinate.longitude] stringValue];
    destlat = [[NSNumber numberWithDouble:selectedHospital.location.coordinate.latitude] stringValue];
    destlong = [[NSNumber numberWithDouble:selectedHospital.location.coordinate.longitude] stringValue];
    //fbuserid = [NSNumber numberWithLong:10208946705766068];
    fbid = [fbuserid stringValue];

    NSString *param1 = @"userid=";
    NSString *param2 = @"&srllat=";
    NSString *param3 = @"&srllong=";
    NSString *param4 = @"&destlat=";
    NSString *param5 = @"&destlong=";
    NSLog(@"%@", srclat);


    //start download
    NSString *requestBody = [[[[[[[[[param1 stringByAppendingString:fbid] stringByAppendingString:param2]
        stringByAppendingString:srclat] stringByAppendingString:param3] stringByAppendingString:srclong]
        stringByAppendingString:param4] stringByAppendingString:destlat]stringByAppendingString:param5]
        stringByAppendingString:destlong];

    [suggestRouteRequest setHTTPBody:[requestBody dataUsingEncoding:NSUTF8StringEncoding]];

    NSData *result = [NSURLConnection sendSynchronousRequest:suggestRouteRequest returningResponse:nil error:nil];
```

RESPONSE FROM SERVER

```
        //creating json object
        NSArray *jsonArray = [NSJSONSerialization JSONObjectWithData:result options:NSJSONReadingMutableContainers error:nil];
        if (!jsonArray) {
            NSLog(@"Error parsing JSON: ");
        } else {

            for(NSDictionary *item in jsonArray) {
                UserRoute *userRouteFromServer = [[UserRoute alloc]init];
                userRouteFromServer.userid = [item valueForKey:@"user_id"];
                NSNumber *srclat = [item valueForKey:@"src_lat"];
                NSNumber *srclong = [item valueForKey:@"src_long"];
                NSNumber *destlat = [item valueForKey:@"dest_lat"];
                NSNumber *destlong = [item valueForKey:@"dest_long"];
                NSNumber *userlat = [item valueForKey:@"user_lat"];
                NSNumber *userlong = [item valueForKey:@"user_long"];
                NSString *timestampString = [item valueForKey:@"timestamp1"];
                userRouteFromServer.tripId = [item valueForKey:@"trip_id"];
//                NSLog(@"%@",timestampString);

                NSNumberFormatter *f = [[NSNumberFormatter alloc]init];
                f.numberStyle = NSNumberFormatterDecimalStyle;

                userRouteFromServer.timestamp =[f numberFromString:timestampString];
                userRouteFromServer.sourceLocation =[[CLLocation alloc]initWithLatitude:[srclat doubleValue] longitude:
                    [srclong doubleValue]];
                userRouteFromServer.destinationLocation =[[CLLocation alloc]initWithLatitude:[destlat doubleValue] longitude:
                    [destlong doubleValue]];
                userRouteFromServer.userLocation =[[CLLocation alloc]initWithLatitude:[userlat doubleValue] longitude:
                    [userlong doubleValue]];

                [userRouteListFromServer addObject:userRouteFromServer];
            }
        }

    }

    return userRouteListFromServer;
}
```

FUTURE SCOPE

1. We will implement the wait time and ETA on the annotations.
2. We will implement the Flip view feature which will show the list of hospitals on map and on the flip side we will provide the list of hospitals on table view with the wait time.
3. Implementation of login feature with username and password.