DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

Summer Research Internship Report

Maulik Chaudhary (201501435)
Vedant Gadhvi (201501135)
Jay Shah (201501071)
Prof. Lavneet Singh

Project problem:
Our project was to create a cross-platform web app for tourists to make their life easier. Tourists will be given information of the site they are are currently visiting (without explicitly pressing any button i.e. on app launch) so they are not exploited by local guides. They will also be given information regarding spots near by their location i.e. right, left. However, if they are not on the location of any tourist site we will be showing possible attractions near(inside a fixed radius of current location of the user) by their current location.

Motivation:
All three members of our group had some background in the field of development. Maulik had prior knowledge of HTML and CSS. Jay knew the basics of JavaScript. Vedant knew some basics of Android development. Also when we discussed the project with Prof. Lavneet Singh, mentor of the project, we were intrigued by his idea of Virtual Tourist Guide. It was a very unique idea in the sense that there are not many application working for tourists. The technology we used was however different from anything we knew but our knowledge surely helped.

Application:
The main application of our project is to provide virtual tourist guide to the tourists. They will get every small detail of the site on their mobile screens which is generally known by real tourist guides. It will minimize work done by the user as it will show the site details just by launching the app without any user interaction, provided the user is inside any site.

Literature Search:
Cross-platform web app means an app that would be supported on both the platforms namely Android and iOS. We initially did some research if there are any systems currently working in this field. Less to our delight we didn't found any such system. We read some online articles which helped us in recognising tourist problems. We had choices between many softwares to create such an app. React-Native, Xamarin, PhoneGap are to name a few. We finally decided to go with React-Native as it is one of the most widely accepted in industry. The language used for React-Native is JSX, which is just an advanced and complicated version of JavaScript (JS). We did an online course on React-Native available on Udemy. We also studied from relevant sites about the technology.

System Architecture:

To properly manage our project we used Visual Studio Team Services (VSTS). In VSTS, we work in the form of iterations. After every iteration we are supposed to complete some assigned work. Every iteration consisted of user stories assigned to members of the team which has to be completed in some predefined time with some priority assigned to it. A new user story has three stages new, active and closed. Each user story has use cases which should be completed by assigned member and it can store research work or code which could be useful for future work. The first iteration consisted of requirement gathering and learning new technologies. During the first iteration we learnt new technologies like HTML, CSS, JavaScript, React-Native. The second iteration consisted of typical software engineering approaches like use cases, sequence diagram corresponding to every use case along with UI design for the MVP(Minimum Viable Product). MVP is a product with just enough features to satisfy early customers, and to provide feedback for future.

We created three use cases:

- Launch Experience
- Onsite Navigation
- Position Accuracy.

Launch Experience: When the app is launched the system will check if the user is onsite or not. onsite means that the user is inside any Site registered on the app. If the user is on site, the information about the site will be shown to the user. And if the user is not onsite he will be shown all the sites in a specified radius of his current location. If the user clicks on any site it's information will be shown to the user.

Onsite Navigation: Once the user is on the site, the site can further be classified into spots present in that site. If user navigates near any spot inside the site the spot's details will be shown to the user. The user will be shown four buttons left, right, front and back. And each button when clicked will be showing the corresponding spots and if there are no spots in a particular direction the corresponding button would be disabled..

Position Accuracy : When user location has some error in gps location and at that location multiple spots are mapped to a single location, user can click on any of the given spot to fetch it's detail or scroll to any other spot shown.

We also created sequence diagrams corresponding to each use case: A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the use case and the sequence of messages exchanged between the objects needed to carry out the functionality of the use case.

Use Case 1: Launch Experience	Pre-condition: App is launched by the user Basic flow: For Onsite · Location of user is obtained-> location of the user is found to be on any site -> show detailed description about that site. Basic flow: For Offsite · Location of the user is obtained-> user is not in near any site-> show him the closest site on the map-> if the user clicks on the site, show its details and directions.
Use Case 2: Onsite Navigation Exp.	Pre-Condition: User is on a tourist site Basic Flow: · User is given four buttons from which he can choose any direction-> after choosing the direction he will be shown all the spots corresponding to that direction. Alternate Flow: · If there are no spots in any direction, the button corresponding to that direction will be disabled.
Use Case 3: Location Precision	Pre-Condition: User has reached the spot Basic Flow: · Once the user has reached the spot, if there are multiple spots on the same location show all the spots-> if user clicks on any spot its details will be shown.

Use Case 1: OnSite

Presentation Layer: When the user Launches the app OnLaunch() method will be called. After loading spots user will be shown the details of site via ShowSiteDetails() method.

Domain Layer:
The OnLaunch() method will call the GetUserCord() method present in infrastructure layer.
After obtaining coordinates of user GetSpotDetails() method will be called.

Infrastructure Layer:
The GetUserCord() method will then communicate with APIs and obtain user coordinates.
GetSpotDetails() will then check the location from the database.
If the location matches from the database load all the spots in the site within a 100 meter distance.

Use Case 1: OffSite

Presentation Layer:
When the user Launches the app OnLaunch() method will be called.
If there are no matches with the database, load the map and show nearby tourist sites.
If user clicks on specific site then details will be shown through ShowDetails() method.

Domain Layer:
The OnLaunch() method will call the GetUserCord() method present in infrastructure layer.
After obtaining coordinates of user GetSpotDetails() method will be called.

Infrastructure Layer:
The GetUserCord() method will then communicate with APIs and obtain user coordinates.
GetSpotDetails() will then check the location from the database.
If there are no matches with the database, load the map

Use Case 2: OnSite Navigation Exp.

Presentation Layer: The user will be shown four buttons forward, backward, right and left, from which he can select button corresponding to that direction. When user clicks any button SearchSpots() method will be called. After SearchSpots() method ShowSpots() method will show every spot to the user corresponding to that direction.
Domain Layer: SearchSpots() will search for every spot in the database corresponding to that direction.
Infrastructure layer:

Use Case 3: Location precision:

Presentation Layer: Sometimes the system cannot differentiate between multiple spots precisely which results in showing multiple spots corresponding to the same location. When the user reaches such a spot he will be shown all the spots corresponding to that location through the method showMultipleSpots(). When user clicks any spot GetSpotDetails() method will be called. User will be shown details of the spot obtained from the GetSpotDetails().
Domain Layer: GetSpotDetails() method will give spot details.

Woking of the system and algorithms :

Firstly we started creating the first use case i.e. Launch Experience. For launch experience we need to access the current location of the user and check in our database if that user is inside any registered site or not. To access the current location navigator.geoLocation API is used. We also need to give permission to access location inside the manifest file of android. First we need to configure geoLocation then we need to authorize it and then we will be able to access latitude and longitude if the current location.

Then we also need to create a real time database to store the values of sites and spots inside the site. For each site we stored its name, Coordinates, opening and closing time, description and images. Inside each site there were multiple spot objects. Each spot object individually had its name, coordinates, description and images. We used Firebase to create our real time database. We initially stored values of sites and spots through a JSON file. We divided sites on the basis of their coordinates in such a way that they lie inside a particular region. A region is defined by 4 points in a rectangular fashion eg. (1,2),(1,5)(5,2)(5,5). We will then check the coordinates obtained from the API and compare with every range. Once the range is obtained we will then compare with the current location. If we find a match with the location in the database the user is considered to be on site or else its off site.

And if the person is off site, we will load the map. We used Google map API for showing map in our App. We require a key for the Google map API. It can be obtained from Google developer site. To obtain map of a particular region we give coordinates of the place to MapView is an inbuilt component in React-Native which helps in rendering map on the screen. The user can then click on any of the sites and the description along with images will be shown.

The second use case is On Site Navigation: Once the person is on any registered site he can then navigate inside the site. The location of the user will be constantly accessed. If the user comes near any spot, description of that spot will be shown to the user. Secondly, once the user comes near any spot four buttons namely right, left, front and back will be enabled. Once any of the four buttons is clicked, names of the spots present in that direction will be shown. We here basically compares the current user coordinates with our database and show the spots accordingly. If let’s say the user wants to visit spots on the left direction, he will start walking towards left direction and once he reaches near any spot the description of that spot will be shown automatically on the screen.

The third use case was Position accuracy: It could happen that on the same location there may be multiple spots registered as we are considering only 5 decimal places for saving coordinates. In such scenarios, we collect the data of all the corresponding spots and present to the user through a scroller. The user can scroll from left to right to get information of the spots.

Future directions: We were thinking to help users to plan their entire day according to the visiting hours of the site. We would give them a schedule which will help them in visiting sites of a city.
We can navigate the user to a particular location. For example, a spot is being shown on the screen of the user along with the four button. The user will click on the right button we will show list of spots on the right of the current spot. If the user selects on a spot, the four buttons will be enabled-disabled to take the user to the selected spot.
Some sites have many spots inside like small temples, we can show the user a preferred path to visit each spot using machine learning. We can obtain dataset by tracking the path preferred by previous users.
We can also improve our app by showing amount of rush on any spot inside the site which can help the user in visiting spots. We can have our person at that site which will give us information about the rush in any spot.

Limitations: The location provided by google API may not be always accurate.
Our system is information oriented. Right now we are providing the information of sites. But in the long run we need to depend on other sources for obtaining it. This rise questions about authentication of the information.
At some places like temples a mobile phone is not allowed. This will destroy the entire functionality of the app.
Rarely it could happen that there is not any path from the left to go to spot showing in the left of the spot.

References: Udemy: Complete course on React-Native and Redux
https://github.com/jacklam718/react-native-popup-dialog/issues/51
https://www.facebook.com/752862324794394/posts/1806827726064510/
https://gist.github.com/nazrdogan/87c63e89a2bfd4cefee24990e4e7ed0e
https://www.youtube.com/watch?v=s23HPMdifvI&t=209s
https://github.com/facebook/react-native