Mobile Tour Guide

Team 2

Doha Moamina, Ali Abod, Adam Babs, Michael Borman, Martin Pugh, Jakob Ropers

Table of Contents

1. Summary of Proposal

1.1 Background, Aim and Objectives

The aim of this report is to describe and document the fundamentals of the tour guide mobile application. In this report we provide a precise description of methods and plans used, that will allow us to more efficiently deliver the final product and avoid mistakes throughout the whole project.

1.2 Project Accomplishments and Progress

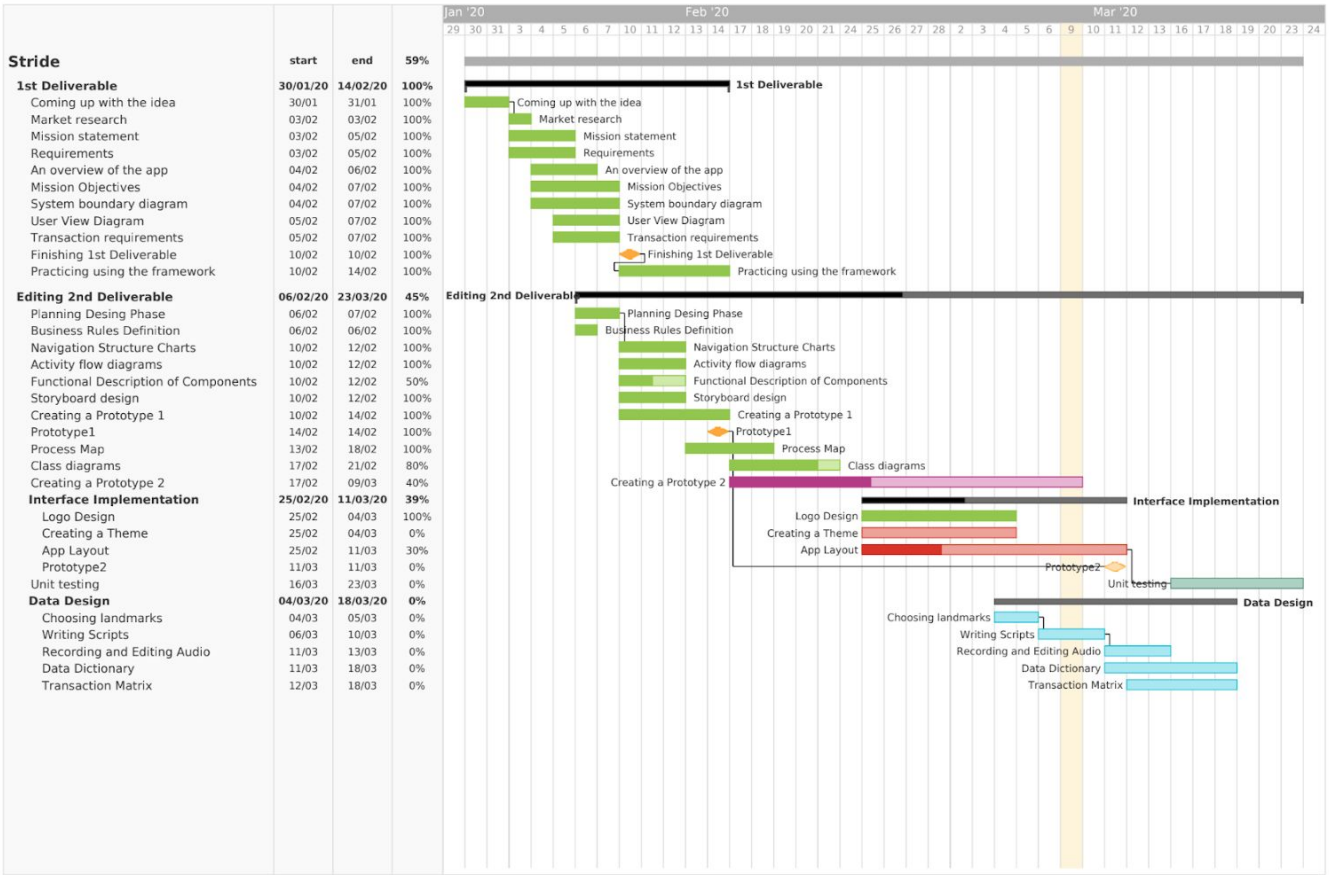


Figure 1

The diagram shown in **Figure 1** is the current gantt chart which was modified slightly to suit the current schedule of each individual member which shifted as the semester progressed. Some of these changes include increasing the time needed to create the second prototype, this was due to the demand for

the developing tool used which has a learning curve that the group members saw as time consuming. We are aspiring to have a working interface prototype within three weeks from the initial starting date.

Now putting changes aside, the progress of the tasks is going ahead smoothly by the group, the charts required in this deliverable were successfully completed and can be seen coloured in green symbolising that they are complete, and the rescheduled second prototype is in pink meaning that it is currently in progress in conjunction with the first deliverable to aid with completing the implementation successfully.

Github has been adopted for task management where the group created a project and a Kanban board on the Github project for that. Kanban is a task management created to circulate tasks as cards between to-do, and finished. New tasks can only be added if all unfinished tasks are complete.[1] This can be used in software development to manage the completion of required functionalities to be implemented, as well as helping members track the progress of other members, which is why it was the chosen management tool. See **Figure 2**.

Overall the progress of the tasks is going smoothly with minor unexpected disturbances such as technical difficulties which lead to rescheduling some tasks.

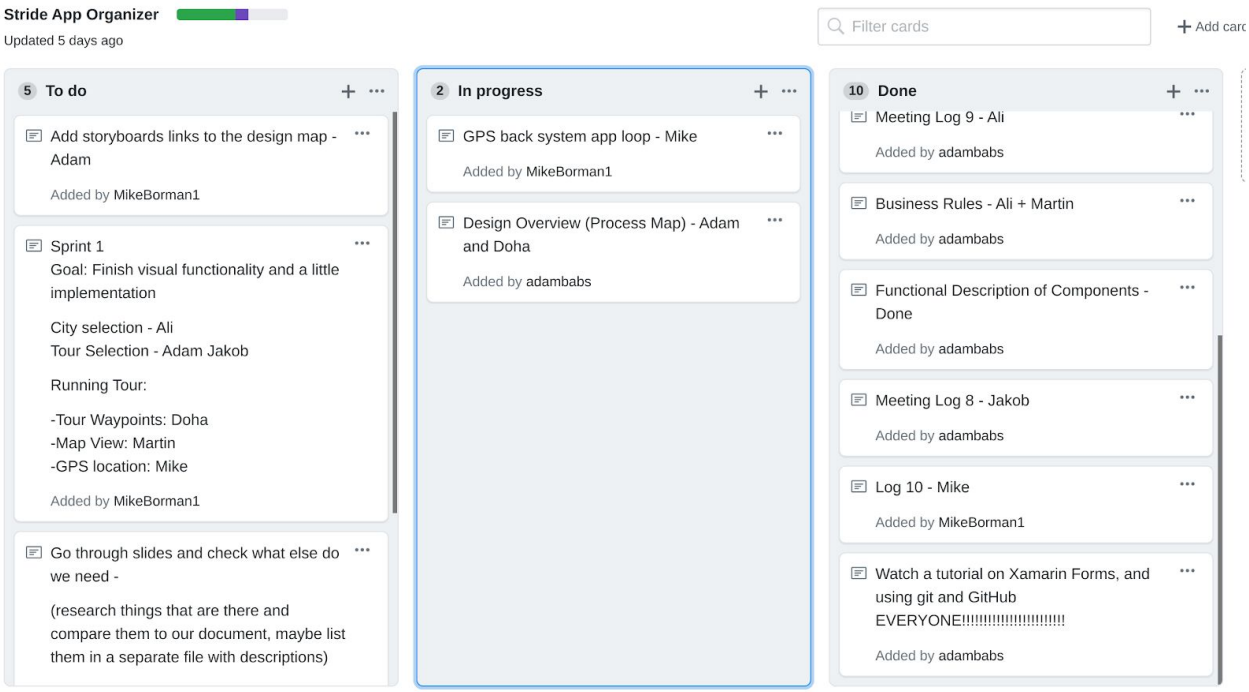


Figure 2

[1] David J. Anderson, *Kanban: Successful Evolutionary Change in Your Technology Business* (Sequim, WA: Blue Hole Press, 2010))

1.3 Framework Choice

To implement the designed application we decided to use Xamarin.Forms. It is a framework framework from Microsoft for building iOS, Android, & Windows apps with .NET from a single shared codebase. This framework is widely used in the Application Development industry, with many features and industry standards, therefore it seemed to be a reasonable choice that will facilitate developing our mobile tour guide.

This framework uses XAML (which stands for "eXtensible Application Markup Language") and C#. The main advantage of this framework is that it allows for cross-platform development on IOS and Android simultaneously, which allows this application to be accessible for a more user-base. [2]

2. Design

This section contains all the diagrams corresponding the the design of the application in question,

2.1 Navigation Structure

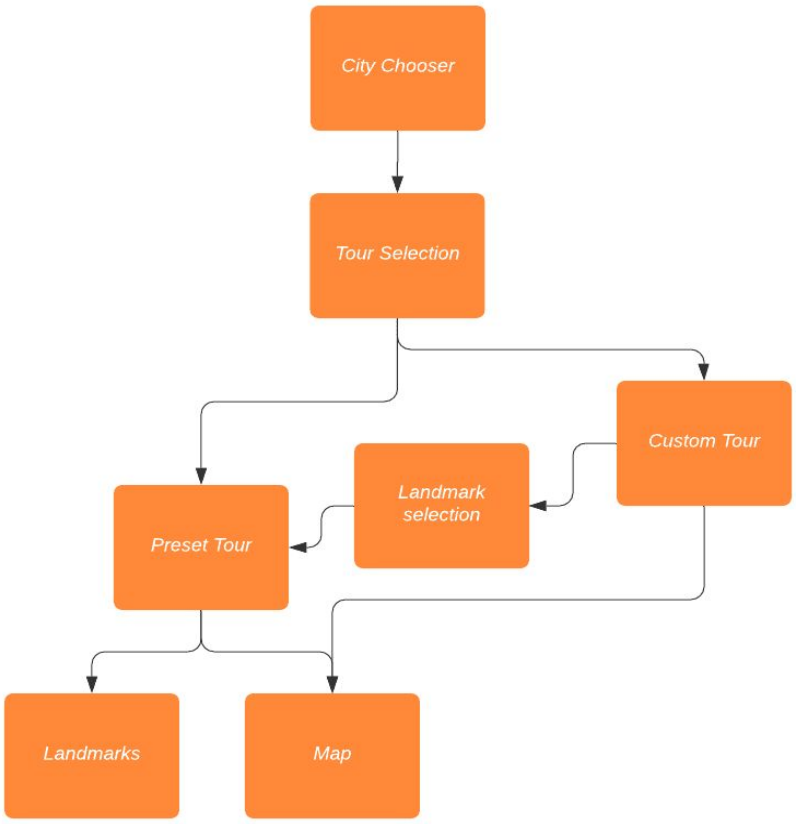


Figure 3

[2] Docs.microsoft.com. 2020. *What Is Xamarin.Forms? - Xamarin*. [online] Available at: <https://docs.microsoft.com/en-us/xamarin/get-started/what-is-xamarin-forms> [Accessed 12 March 2020].

The user's journey begins with selecting a city. This is the same for all users, whether they are regular users or are first time users. Each city has a selection of tours which the user is presented with on the next navigation node. This is in a hierarchical structure.

Users that select a preset tour are then presented with the tour page. This is made up of two tabbed modules, the map and the landmarks. The default is the map module, but once the landmarks button is tapped the user is presented with a list of the landmarks on the tour. These modules are in a flat navigation structure.

Users that select the custom tour option proceed to a page that allows them to select their landmarks for the tour. There is also a tab module on the page that presents the map to the user to allow them to see the landmark locations. The map and landmark selection are in a flat structure. Once landmarks are selected the user proceeds to the same page as a preset option and are able to begin their tour.

2.2 Activity-Flow Diagram

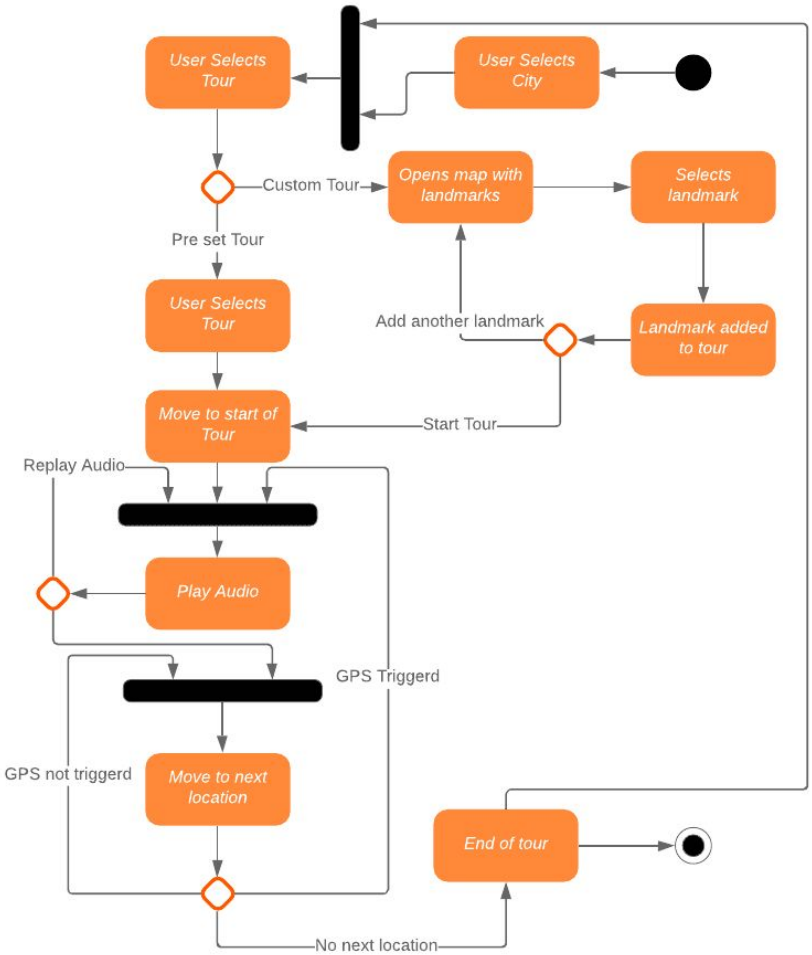


Figure 4

Either at the start of a new tour or at the end of a previous tour, a user can select a city, then select a tour. The user then has the option to select either a pre-set tour or a custom tour. If the user selects a custom tour, then a map with landmarks is opened where the user can select the landmarks they wish to visit. The landmark is then added to the tour. The user then has two more options: start the tour or add another landmark. If the user selects a premade tour, then the tour starts. Once the tour is started, the GPS is triggered, and the audio is played. Once the next location is reached, the next audio navigation begins. When there is no next location, the tour ends.

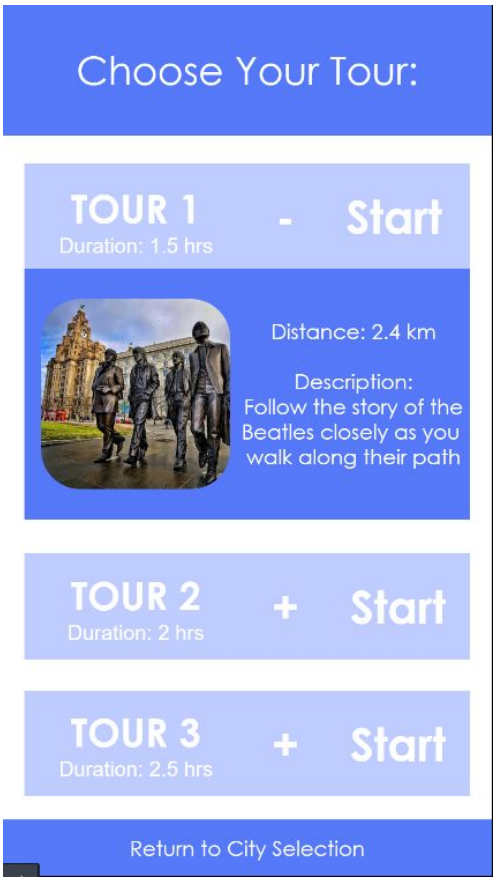
2.3 Storyboards

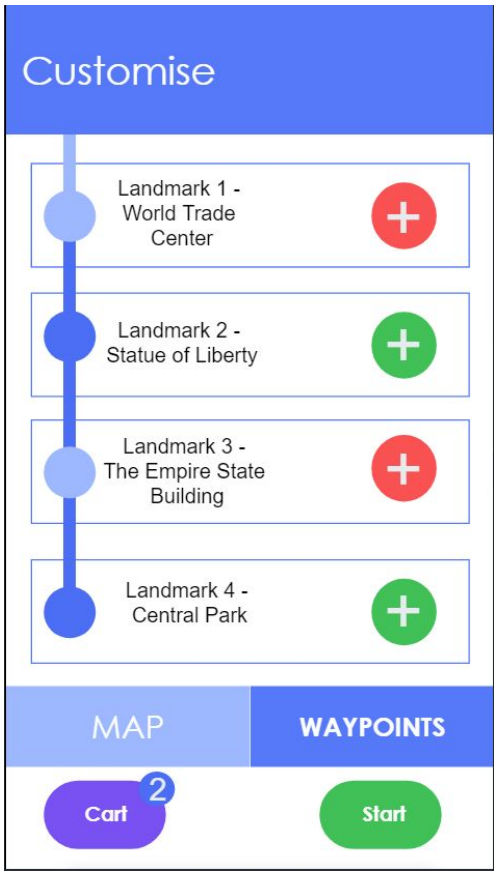
View 1 **View 2**

City Selection (View 1): This view presents the available cities where the user can start a tour with Stride. This is the first view when starting the application, and the user is also returned to this view once he/she completes or stops a tour. Each city name and its image work as a button that will take the user to the **Tour Selection** (View 2) page for the selected city. This view is related to **Use Case #1 ("Select City")**.

Tour Selection (View 2): This view presents the user with the tours available in the city he/she selected in the previous view (View 1). Each tour element can be expanded when clicking the "+" symbol, which

will take the user to the **Expanded Tour Selection** view (View 3). Additionally, the user can press the start button to start a tour, taking him/she to the running tour views (View 5-7) unless the "Custom Tour" is selected, which will take the user to the **Tour Customization** view (View 4). Finally, the user can click "Return to City Selection" to move back to the **City Selection** view (View 1) and change the city selection. This view is related to **Use Case #2 ("Select a Tour")**, **Use Case #4 ("Start Tour")**, and **Use Case #6 ("See Information on Tour")**.



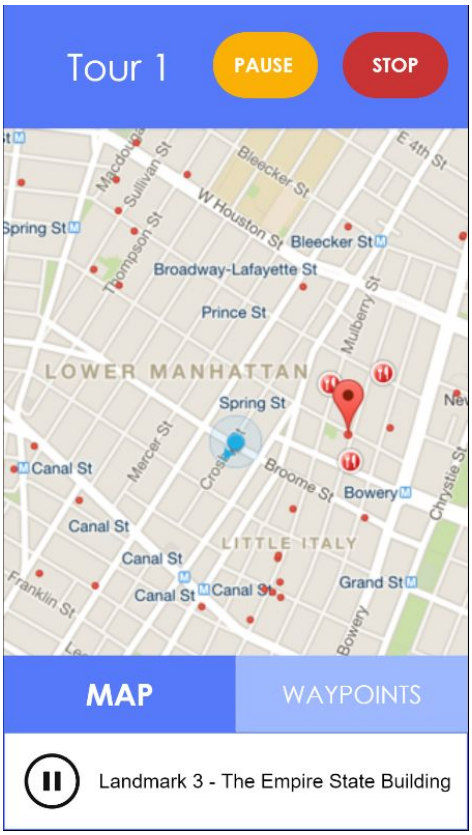


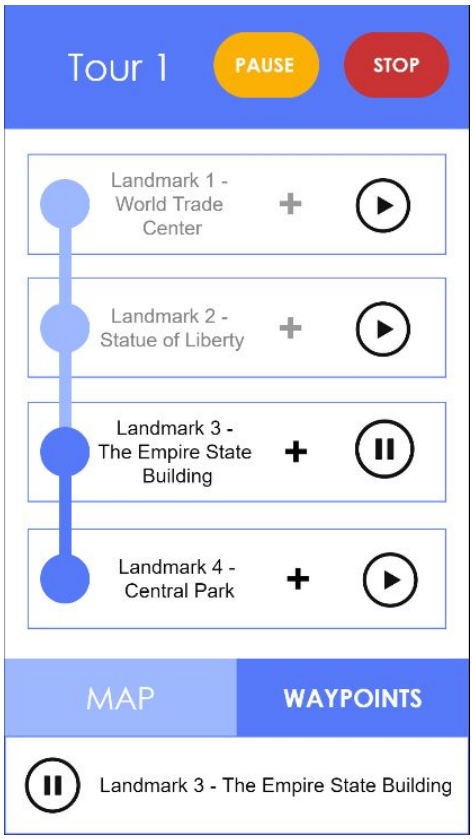
View 3 **View 4**

Expanded Tour Selection (View 3): A expanded view of View 2. This shows the user an image representing the theme of the tour as well as some additional information on the tour. When the user clicks the "-" symbol, the view will return to **Tour Selection** view (View 2). In addition, when clicking "Start", the same will happen as described for **Tour Selection** (View 2). This view is related to **Use Case #2 ("Select a Tour")**, **Use Case #4 ("Start Tour")**, and **Use Case #6 ("See Information on Tour")**.

Tour Customization (View 4): This view allows the user to create his/her own tour. By default, the "Cart" will be empty when entering this view. The user will see a list of all the landmarks available in the city he/she selected in View 1. When the cart is empty, all "+" symbols will be marked as red. When the user wants to add a landmark to the tour, he/she must click the "+" button, resulting in the tour being

added to the list, the blue marker next to the landmark name becoming dark blue, the color of the "+" button changing to green, and finally, the number at the "Cart" updating to the total number of landmarks in the cart. Once the user clicks "Start", he/she will be taken to the running tour views (View 5-7). This view is related to **Use Case #3 ("Customize Tour")** and **Use Case #4 ("Start Tour")**.



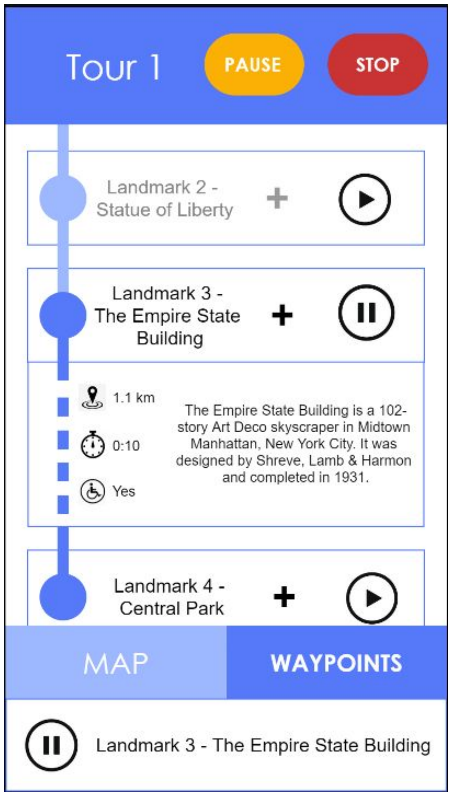


View 5 **View 6**

Running Tour Map View (View 5): This view shows the user the map for a running tour. This view will show the route the user has to walk along, his/her own location, as well as the landmarks that are part of the selected tour. The user can click the "Pause" at the top to pause the tour. This will save progress and put the tour on hold. The button will change to "Play" and the user can press this again to continue the tour. In addition, the user can press the "Stop" button to quit the tour, returning him/her back to the **City Selection** view (View 1) and saving no progress. The user is also able to move the map around to check out the route and landmarks. Also, the user can click the "Waypoints" tab button to move to the **Running Tour Waypoints View** (View 6).Finally, the user can use the "||" (Pause) Symbol in the bottom to pause the audio. This will change the button to a "▷" symbol, which, if pressed, will continue playing the audio again. This view is related to **Use Case #5 ("Pause Tour")**.

Running Tour Waypoints View (View 6): This view shows the waypoints of a running tour. If a landmark has already been visited, it will fade out like "Landmark 1" and "Landmark 2". The landmarks are in the order they will be visited. One can play and pause the individual audio of each landmark at any time. Only one audio can be played at a time. One can also press the "+" button to move to **Running**

Tour Waypoints Expanded View (View 7). One can also press the "Map" tab button to move back to the **Running Tour Map View** (View 5). Other than that, all other controls are as described for View 5. This view is related to **Use Case #5 ("Pause Tour")**, **Use Case #7 ("Play audio for Landmark")**, **Use Case #8 ("See Information on Landmark").**



View 7

Running Tour Waypoints Expanded View (View 7): This is the expanded view of View 6. All functionalities are the same as described in View 6. This view gives the user additional information on the landmark. If the "+" symbol is clicked again the app will move back to the **Running Tour Waypoints View** (View 6). This view is related to **Use Case #5 ("Pause Tour")**, **Use Case #7 ("Play audio for Landmark")**, **Use Case #8 ("See Information on Landmark").**

2.4 Design Process Map

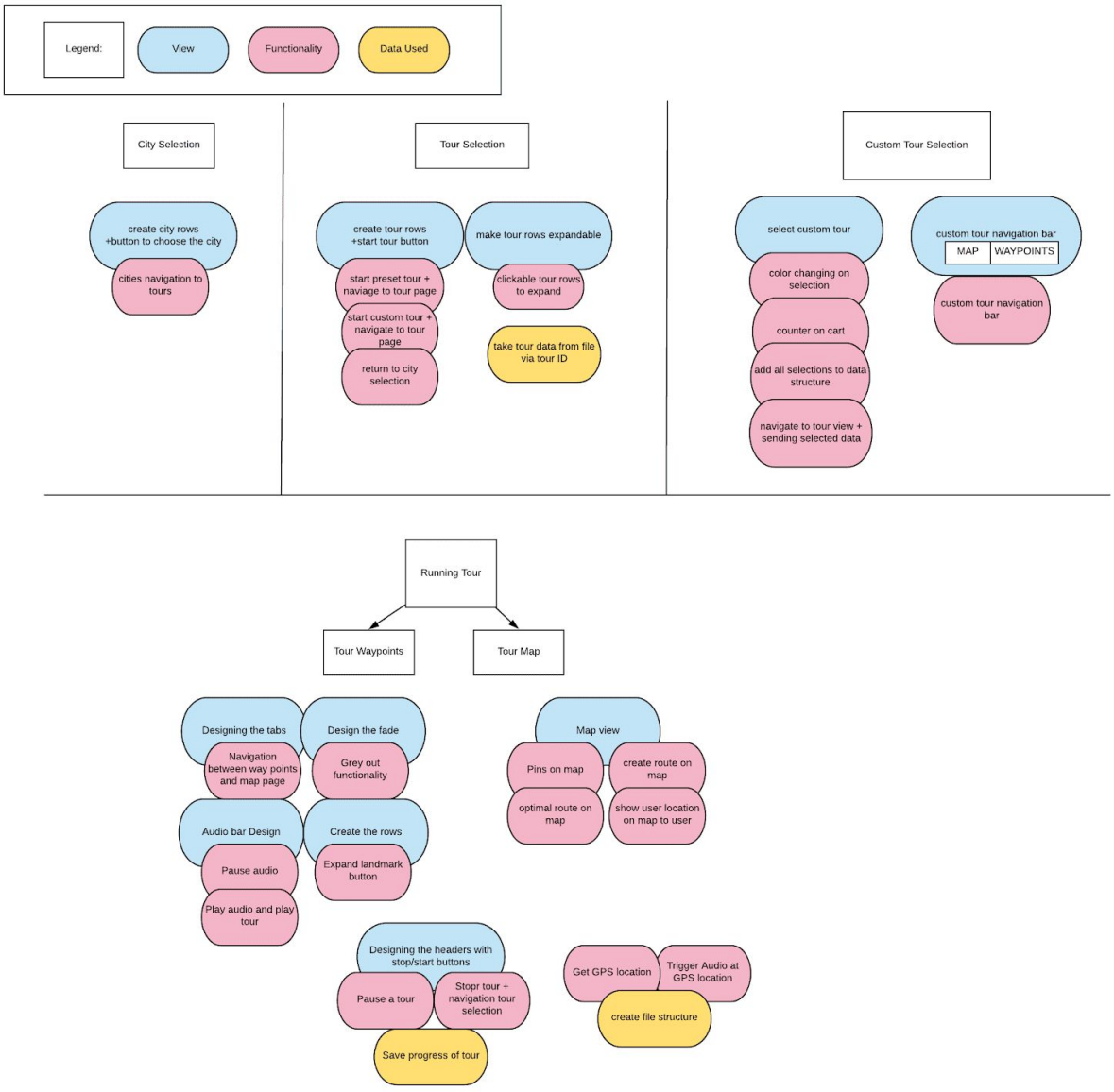


Figure 5

2.5 Anticipated Functional Components of the System

As it can be seen in **Figure 5**, we included many functional components represented with the pink rectangles, all of them are discussed in this section.

- Data storage - As will be stated in more detail we will need a place to store the latitudes and longitudes of points of interest and also the tours which will be made up of a subset of points in a specific order. While in reality this should be stored database wise, we have decided initially to achieve this via csv to save on development time.

- Data Retrieval - We will need a set of classes to firstly retrive the data from the csv's and then format and cast to a certain class. So that the data can be retrieved from any view model and will help with decoupling the system further.

- Expandable objects - We will utilise expandable feature on a few of our objects such as tours and points of interest to display extra data. This simply helps declutter the view allowing the user to understand and navigate through our system efficiently.

- Map View - The map view feature will be an essential part of the running tour, not only showing all waypoints but the route through them. This will be implemented through a maps API(most likely Google Maps) which will reduce complexity on our side. The maps API will also allow us to also direct the user to the beginning of the tour no matter where they are located, giving accurate time predictions for different modes of travel. Furthermore for the custom tour it will automatically find the optimal route between pins given the users current location.

- GeoLocator - An essential part of the system, this feature, implemented with a GeoLocator plugin will allow us to get the current location of the user through different time intervals. This data will be used to update the map regularly, but more importantly to trigger the voice activation files, which will play an audio file when the user is within a certain distance. Time intervals for updating the users location will be within the 5-10 second range. This feature also requires permissions for both IOS and android, which the user will be prompted with.

- Audio activation - There are two parts to this feature, firstly the activation, based either on proximity or through user interaction with the application, and secondly the retrieval and playback of said audio. The audio files will be kept in the same directory as the application and downloaded on installation, however in the future as the file size grows this may need to be done through remote blob storage.

- Main App loop - This is where the core of the logic occurs and is responsible for checking the location against the waypoints locations and triggering the audio and other visual features connected to it. This can be done either in a hard coded time loop which may prove to be inefficient, or through some tertier function such as the notifyPropertyChanged event handler.

2.6 Business Rules

These business rules stand for the terms and conditions that the user shall expect from the developers, and what users should comply with when using the application, and they are as follows:

1. In order to start a tour, users have to agree to using their GPS location.
2. App is only for private use.
3. We will not store users GPS information.
4. We will not store any user's information.
5. Do not distribute our application.
6. Do not provide false information about the business.
7. Do not display distracting content, for example, gibberish, gimmicky character use, misspellings, etc.
8. Do not display links.
9. No offensive or inappropriate content.
10. Published content must not discriminate: gender identity, sexual orientation, gender, age, religion, ethnic origin or disability.
11. No use of obscene, profane or offensive language
12. No sexually suggestive or explicit content.
13. To use the app, users need to consent to terms and conditions.
14. Information may not always be accurate. We use third parties to provide data on which the App's information and recommendations are based. We and our data providers try to make sure that the data is correct and up-to-date, but we cannot guarantee that it will always be. Accordingly, we do not accept any liability for any error or omission in the information available or the recommendations made through the App, and we exclude all liability for any action you may take or loss or injury you may suffer (direct or indirect including loss of income, profit, opportunity or time) as a result of relying on any information available through the App.
15. We do not require you to create an account when you download the app. This is because we do not collect data. Any custom tours created will be stored locally and it is volatile (upon closing the app, all data will be removed).

2.7 UML Class Diagram

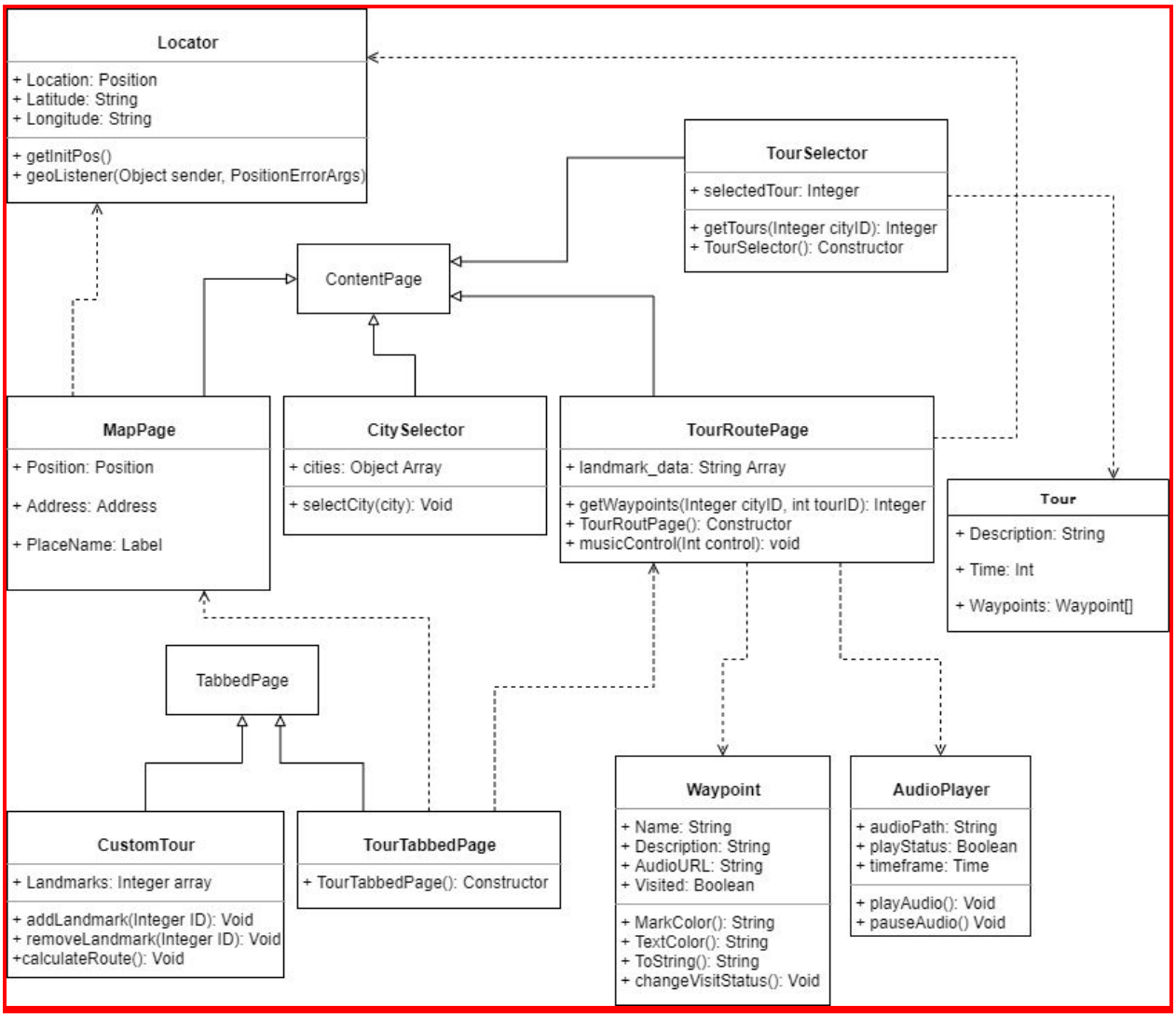


Figure 6

2.8 Use Cases

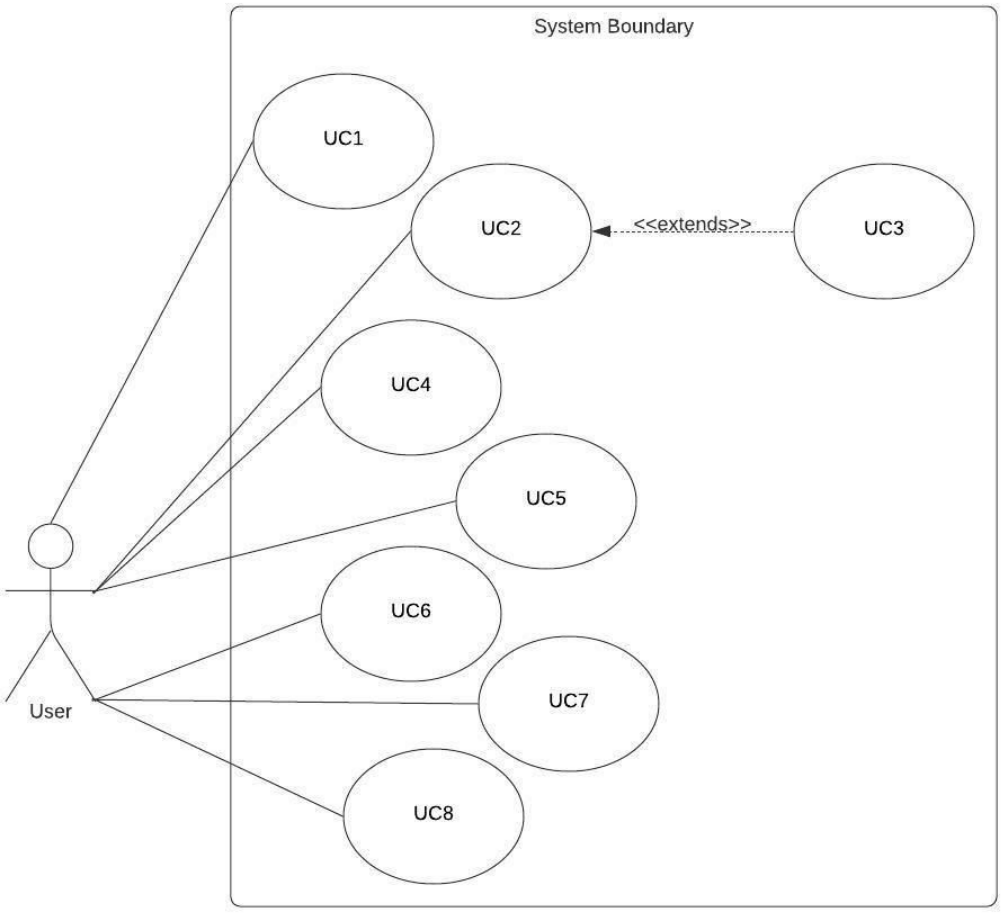


Figure 7

Use Case Descriptions:

ID	UC1
Name	Select City
Description	The actor selects the city in which he/she would like to view the tours for.
Event Flow	The actor views the list of cities, clicks on one, and confirms this choice to select the city.
Triggers	1. The first time the app opens 2. The actor clicks the button to select a city
Post-Condition	A city has successfully been selected and the tour options for that selection is now displayed.

ID	UC2
Name	Select a Tour
Description	The actor selects one of the tour-options.
Pre-Condition	A city has been selected·
Event Flow	1. The actor selects a preset tour which starts right away 2. The actor selects a custom tour which takes him to the tour customization screen
Extension Points	UC3: “Customize Tour”
Triggers	1. UC1: “Select City” 2. The actor clicks the button to select a new tour
Post-Condition	The tour that the actor created has been calculated, predicted distance and duration are displayed, and the tour is ready to be started.

ID	UC3
Name	Customize Tour
Description	The actor selects any number of landmarks present in the selected city to create his/her customized tour.
Pre-Condition	The actor chose a custom tour
Event Flow	The actor selects any number of landmarks in any order. The app will then calculate an optimal route to create a custom tour for the actor.
Post-Condition	The tour that the actor created has been calculated, predicted distance and duration are displayed, and the tour is ready to be started.

ID	UC4
Name	Start Tour
Description	The actor starts the tour, meaning that navigation along the route starts and the GPS audio guide is engaged.
Pre-Condition	A tour has been selected or paused.
Event Flow	1. The actor has selected a new tour and starts it to engage navigation and audio guide 2. The actor has a paused tour and starts it to continue navigation and audio guide where he left off.

Post-Condition	The actor can now see the navigation along the selected route, as well as hear the audio guide.

ID	UC5
Name	Pause Tour
Description	The actor pauses the tour, meaning the app will stop navigation and audio guide, but saves all progress to continue
Pre-Condition	A tour is currently running.
Event Flow	The actor clicks “Pause Tour” in order to save the current progress and put navigation and audio guide on hold.
Post-Condition	The tour is paused, meaning that navigation is not running and the audio guide is not running, but progress is saved.

ID	UC6
Name	See Information on Tour
Description	The actor can click on Tours to see how long they will take, how far they are, a short sentence description of the general theme (if available), and the landmarks it will visit.
Pre-Condition	A tour has been selected
Event Flow	1. Preset Tour selected → The actor can click a button to see extra information including duration, distance, description, and list of landmarks. 2. Customized Tour selected → The actor can click a button to see extra information including predicted duration, distance, and a list of landmarks.
Post-Condition	The actor can now see the information on the tour.

ID	UC7
Name	Play audio for Landmark
Description	At any point of the tour, the actor can pick any landmark of the tour and listen to the audio for it.
Pre-Condition	The actor is currently on a tour and selected a landmark to play the audio for.
Event Flow	The actor selects a landmark shown on the list of landmarks for the tour and presses the “Audio” button to listen to the audio for that landmark.

Post-Condition	The audio for the selected landmark is now playing.

ID	UC8
Name	See Information on Landmark
Description	The actor can click on a landmark to see information on it including a description, year of origin, etc.
Pre-Condition	1. The actor is currently on a tour 2. The actor is currently selecting a tour
Event Flow	The actor selects a landmark shown on the list of landmarks for the tour and presses the "Information" button to view the information.
Post-Condition	The information for the selected landmark is displayed.

2.9 Data Structures Used

To navigate around the application, pages and their functions will need information on where the user previously has been and adjust their performance according to it.

It is a very efficient way to control the application and the relations between pages selected by the user

Navigation stack is a professional way of navigating in apps and it is a widely used method in the application development industry.

```
// if user goes to the next page
Push(S,page)
    if Stack-Full(S)
    then error "overflow"
    else top(S) = top(S) + 1
        S[top(S)] = x

// if user chooses to go to the previous page
Pop(S)
    if Stack-Empty(S)
    then error "you can not go back from here"
    else top(S) = top(S) - 1
        return S[top(S) + 1] //return to the popped element (page)

// in the starting page
Stack-Empty(S)
    if top(S) = 0
    then return True
    else return False
```

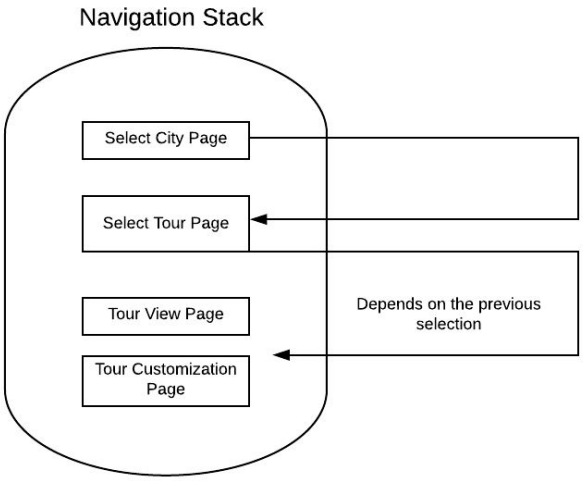


Figure 8

2.10 Our Database and data handling description (or lack of it)

While we do plan to implement a database for our application, we kept this for the future development plan as we want to focus on the actual functionality of the app first. However, we still need some kind of data solution to store our limited tour data for the city of Liverpool and have decided to use three CSV files to store all the data we require. We felt that this solution was efficient enough for the first steps of the app and can then be replaced by a similar database setup for a more scalable solution when spreading the app to more cities across the globe. The three CSV files each represent a table and have primary and foreign keys building relationships between them. The three tables are: Cities, Tours, and Landmarks.

As shown in **Figure 9** below, the **City** table holds a unique *CityID* as its primary key, its name (*CityName*), and finally the location of an image of the city (*CityImage*). Next, we have the **Tour** table, which also holds a unique *TourID* as its primary key, it holds the *CityID* as its foreign key to build a relationship between City and Tour, and it holds its many other attributes storing data on the tour as well as the source for an image representing the tour (*TourName, TourDescription, TourDuration, TourDistance, LandmarkImage*). As shown in the diagram, one city can have many tours and many tours can have only one city. Finally, we have the **Landmark** table, which holds a unique *LandmarkID* as its primary key, and has two foreign keys, *CityID* and *TourID,* to create the Landmark's relationship with the city it is in and the tours it is part of. Again, as shown in the diagram, one or many tours can have many landmarks and many landmarks can have one or many tours they are part of. Also, one city can have many landmarks within it. Finally, the landmark has many attributes to store information about each landmark, including the source for the audio file that will be played when reaching the *LandmarkGPSLocation*.

Overall, the use of CSV files allows us an easy and efficient bridging solution until we have the resources to create a database in future development. The app will be able to easily read data from the files and use this for all its functionalities. The use of primary and foreign keys allows us to easily use the relationships to meet our requirements. When the user selects a city, the **City** table is displayed. Once the user selects a city, all the tours which have the city's primary key as their foreign key will be displayed. Once the user selects a tour, the landmarks holding the tour's primary key as a foreign key will be loaded as part of the tour. Finally, if a user selects the custom tour within a city, all the landmarks holding the city's primary key as their foreign key will be loaded and made available in the selection.

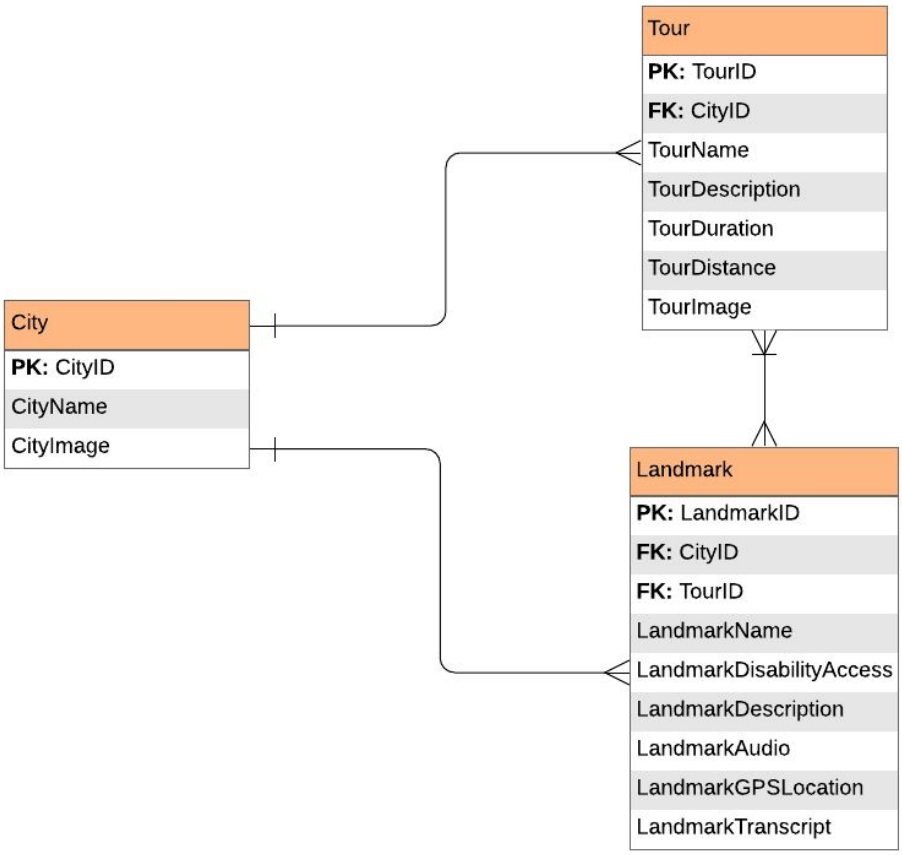


Figure 9

NOTE: PK=Primary Key, **FK**=Foreign Key