COMP 4721 – Software Design
Milestone 1 – Task Identification, Data Flow, and Overall Design
Front End Team (Mariah, Alex, Jane, Filip)
2015-01-30

This document contains a collection of information regarding the design of a geofencing application. The project is still in its early stages. The following sections cover different areas of planning and decisions that have been made so far in the process of designing this app. Two teams are working together to create the front end of the app and the back end of the app, respectively. This document has been compiled by the front end group and refers mainly to information about the front end of the project.

+ ---------- +
+ ---------- +

Team Roles

These are the specific roles each member of the team fulfils. A team member with a given role should be the lead in that role, but all members should of course contribute to each part of the project.

Our group has divided the roles as follows:

- **Mariah Martin Shein:** Team Lead, Architect
- **Alex Francioni:** Documentation Lead, Product Owner
- **Jane Henderson:** Process Co-ordinator, Archivist
- **Filip de Figueiredo:** Quality Assurance, Architect

+ ---------- +
+ ---------- +

Design Log

Overall rationale:

This app is designed for tourists who are interested in the history of the cities they visit. In many old cities, inscriptions have been made on building walls, fountains, and other locations of historic importance. These inscriptions may not be in the local language and are sometimes hard to find. This app will help tourists to locate the inscriptions by sending a notification to their phone whenever they enter the area around the inscription. It will also translate and clarify the inscription with relevant details.

The app is not meant to function as a guided walking tour, but instead to provide friendly, serendipitous notifications that enhance the tourist's visit to the city. As such, the app should be as informative and unobtrusive as possible.

Architectural ideas:

The inscription data for all the cities will be kept in a database on a server that can be accessed by the app via the Internet. Once the user enters a supported city and has their phone connected to the Internet, all the data for that city will be downloaded and stored locally, but not in a way accessible to the user. Once the user enters a particular geofence, they are notified and the corresponding inscription is added to their history. The UI will allow users to view the visited inscription information at their leisure, and keep data from a certain number of previously visited cities.

Guiding concepts:

We would like the app to be helpful but not annoying. We have given a lot of thought to how to handle the notifications so that the user is not constantly harassed by the same inscription that they often pass by, and believe that our design of automatically turning off subsequent notifications and providing a toggle to turn them back on when desired will be effective.

We would also like the app to be useable while not connected to the Internet; however, at the same time, we would not like to use too much space on the user's phone. So we plan to download as much data as we can on to the phone while providing the ability to delete unnecessary data.

+ ---------- +
+ ---------- +

Requirements and Specifications

The initial requirements and architectural specifications are important pieces of groundwork in a project such as this. Our group has outlined various requirements and specifications below.

The initial requirements and specifications of the app are as follows:

- Sensitivity to geolocation
  - Respond to entrance of new geofence by sending notification to user
  - Do not notify user if notifications are toggled off
  - Download new city to local database when city area is entered
- Clear presentation of information
  - Translations can be turned on and off
  - Original inscription is presented so that it can be read more easily
  - Inscription includes a picture to help with location

    - Additional information about location and historical relevance
    - Ability to look through history of inscriptions viewed
- Good app data management
    - Keep all the inscriptions for a visited city in a local database (on the phone)
    - Store inscription data for all possible cities on a server accessed by internet
    - If the local database becomes full, notify user to delete some cities

+ ---------- +
+ ---------- +

Goals

A project of this scale has various goals. It is important to keep track of these goals and set up a timeline for ourselves so as to make sure we are constantly on track. What follows is a list of goals and what each goal entails. Also noted are which goals are more high risk than others.

1) Milestone One/Two/Three
- Complete, hand in, revise based on comments/suggestions

2) Build simple Android Test application:
- Build a simple application functioning to access at least two different views, with at least two buttons

3) Build simple Geofence application:
- Build a simple application implementing the Geofence API
- Ensure notifications are working for various location objects
- Serve as base application to build on

4) Implement a Local Database:
- Implement a local database to store location objects for the city radius downloaded from the main database
- Requires learning how to implement the local database
- Requires testing of local database using various cases

5) Store and Display History
- Implement a data structure to store history
- Requires decision on what data structure is optimal for the task

6) Build user Interface
- XML
- Graphical User Interface (GUI)
- Requires team decisions/research on optimal user interface solutions (UX Design book)

7) Using Inscription Objects
- Requires getting the inscription from the back end team
- Requires team decisions on best ways to display inscription objects:
  - Latin text
  - Translations
  - Picture?
  - Ranking (Easy/Medium/Hard)

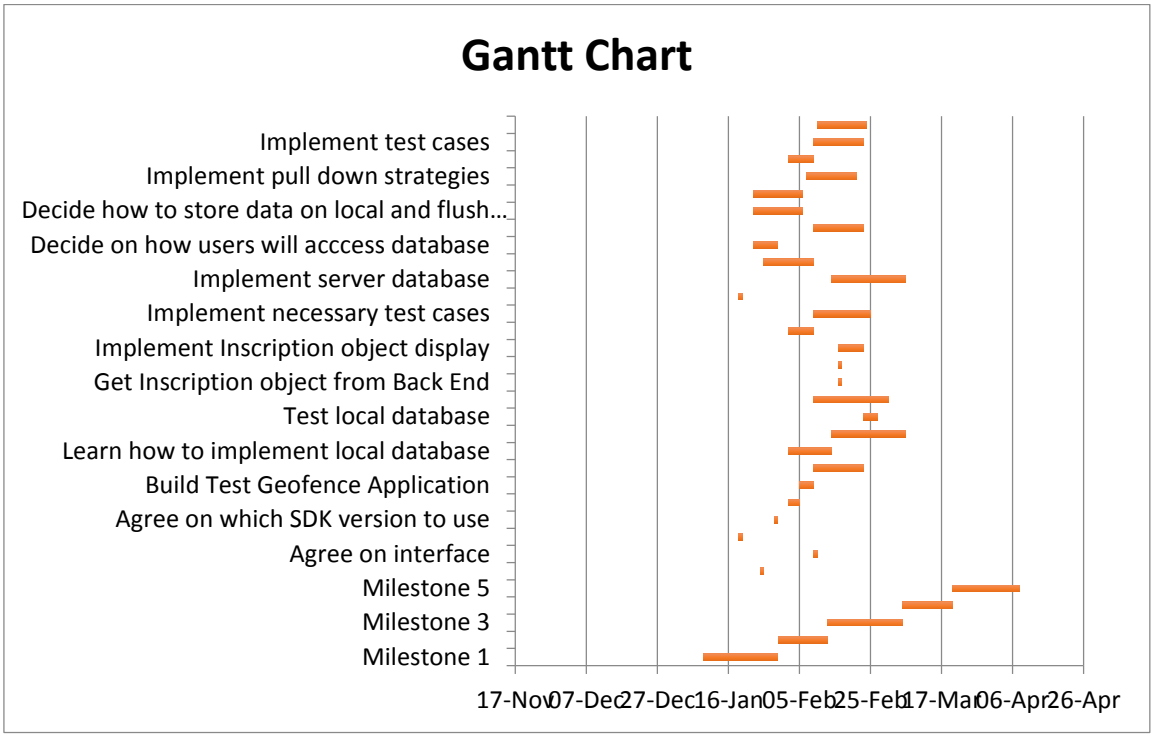
Critical Items:
- Main Database
- Local Database
- Geofence Implementation
- Inscription Objects

Less High-Cost/Risk:
- UX Design
- History Store/Display
- Notifications

+ ---------- +
+ ---------- +

Gantt Chart Timeline

A Gantt-style chart containing our goals, their proposed deadlines, and how they overlap. Done in conjunction with our back end counterpart.



Here is a list of the goals with their predicted times.

Task	Start Date	Duration	End Date
Milestone 1	09-Jan	21	30-Jan
Milestone 2	30-Jan	14	13-Feb
Milestone 3	13-Feb	21	06-Mar
Milestone 4	06-Mar	14	20-Mar
Milestone 5	20-Mar	19	08-Apr
Download Android Studio	25-Jan	1	26-Jan
Agree on interface	09-Feb	1	10-Feb
Setup Git repository (Fron End)	19-Jan	1	20-Jan
Agree on which SDK version to use	29-Jan	1	30-Jan
Build First Test Application	02-Feb	3	05-Feb
Build Test Geofence Application	05-Feb	4	09-Feb
Decide/Build storage of history	09-Feb	14	23-Feb
Learn how to implement local database	02-Feb	12	14-Feb
Implement local database	14-Feb	21	07-Mar
Test local database	23-Feb	4	27-Feb
Build UX	09-Feb	21	02-Mar
Get Inscription object from Back End	16-Feb	1	17-Feb
Decide on Inscription object display	16-Feb	1	17-Feb
Implement Inscription object display	16-Feb	7	23-Feb
Identify necessary test cases	02-Feb	7	09-Feb
Implement necessary test cases	09-Feb	16	25-Feb

+ ---------- +
+ ---------- +

Data Flows

Data flows describe the data requirements of each major module of the project and how these requirements will be satisfied. As the project is still in its early stages, the modules are not yet fully fleshed out, so our data flows may need to be updated or added to in the future.

Our current data flows are as follows:

- The Geofence feature will need to continuously access the data for where the user is currently and reference that with the known inscription locations by accessing the local database.
- When a notification is sent to the user, the associated Inscription object needs to be loaded from the local database and some brief information displayed to the user.
  - This data will then be used when opening the notification to fill out the Inscription Page.

    - At the same time, the user's History will be updated with a reference to the Inscription object.
    - Same process as above (minus the modification of History) will occur when the user visits an inscription through their history page
- When the user opens their History page, the list of known databases will have to be loaded in from the local database that stores History.
    - Toggling any inscription must update the local database to have that inscription turned off.
    - Similar process must occur when visiting the Cities page.
- When the user presses a button to navigate between screens, the screen needs to update to the correct display. Data should be inherent to the class type.
- When deleting a City, the user's History for that data needs to be erased. The local database needs to delete the city from storage as well.

+ ---------- +
+ ---------- +

Work Flows and Use Cases

Work flows and use cases involve how the user interacts with the application. A use case is rather like a subset of a work flow, so use cases are identified with the work flow they are involved with.

We imagined our work flows as a sort of directed graph. Two images depicting our work flows have been provided. These images contain both workflows and data flows – the user's actions are indicated in blue and are where we will take our work flows from. The orange indicates a data flow and will be looked at when examining the use cases.

Charts are on the next two pages, followed by descriptions of the work flows and use cases:

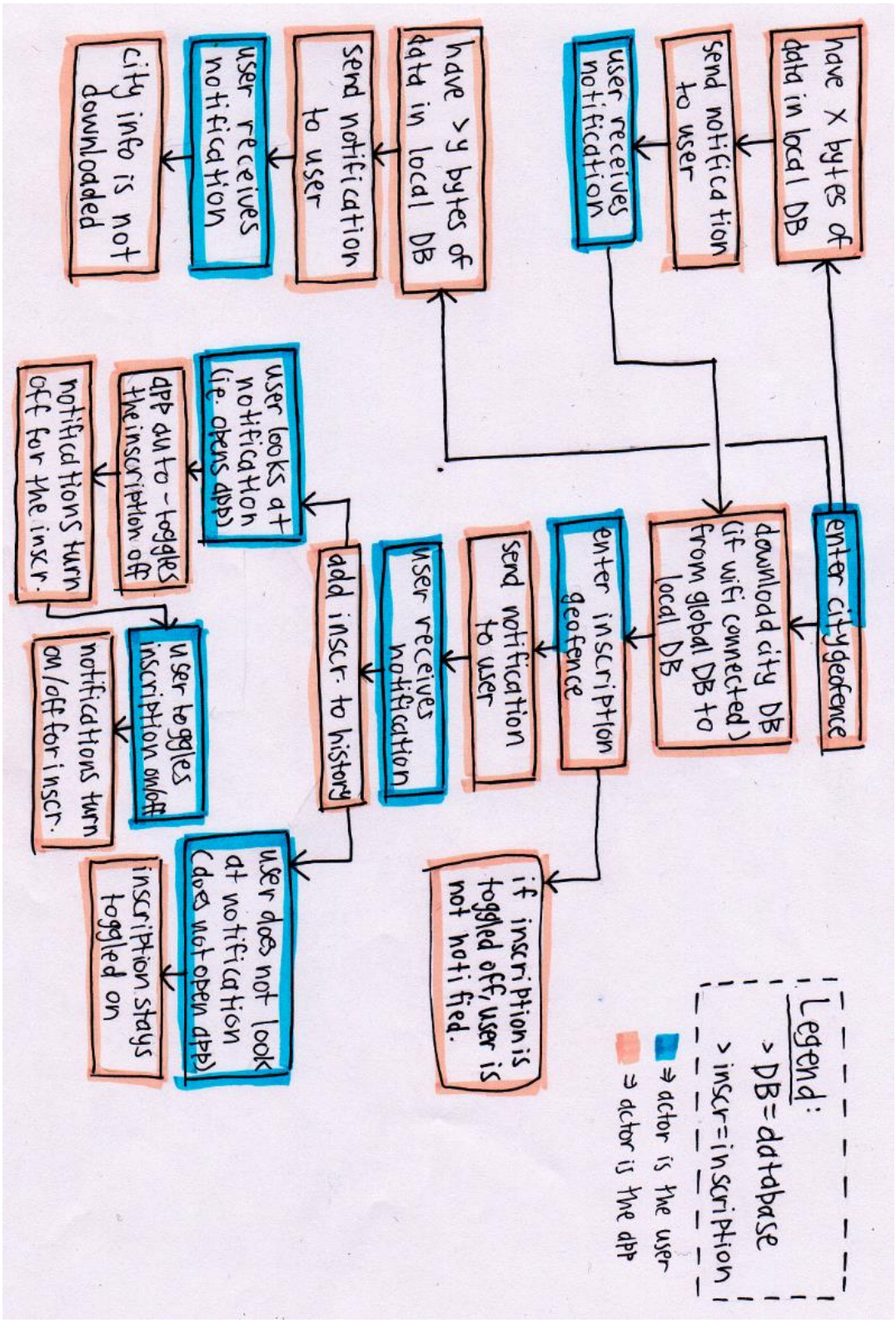


Figure 1. Workflows for entering a geofence area

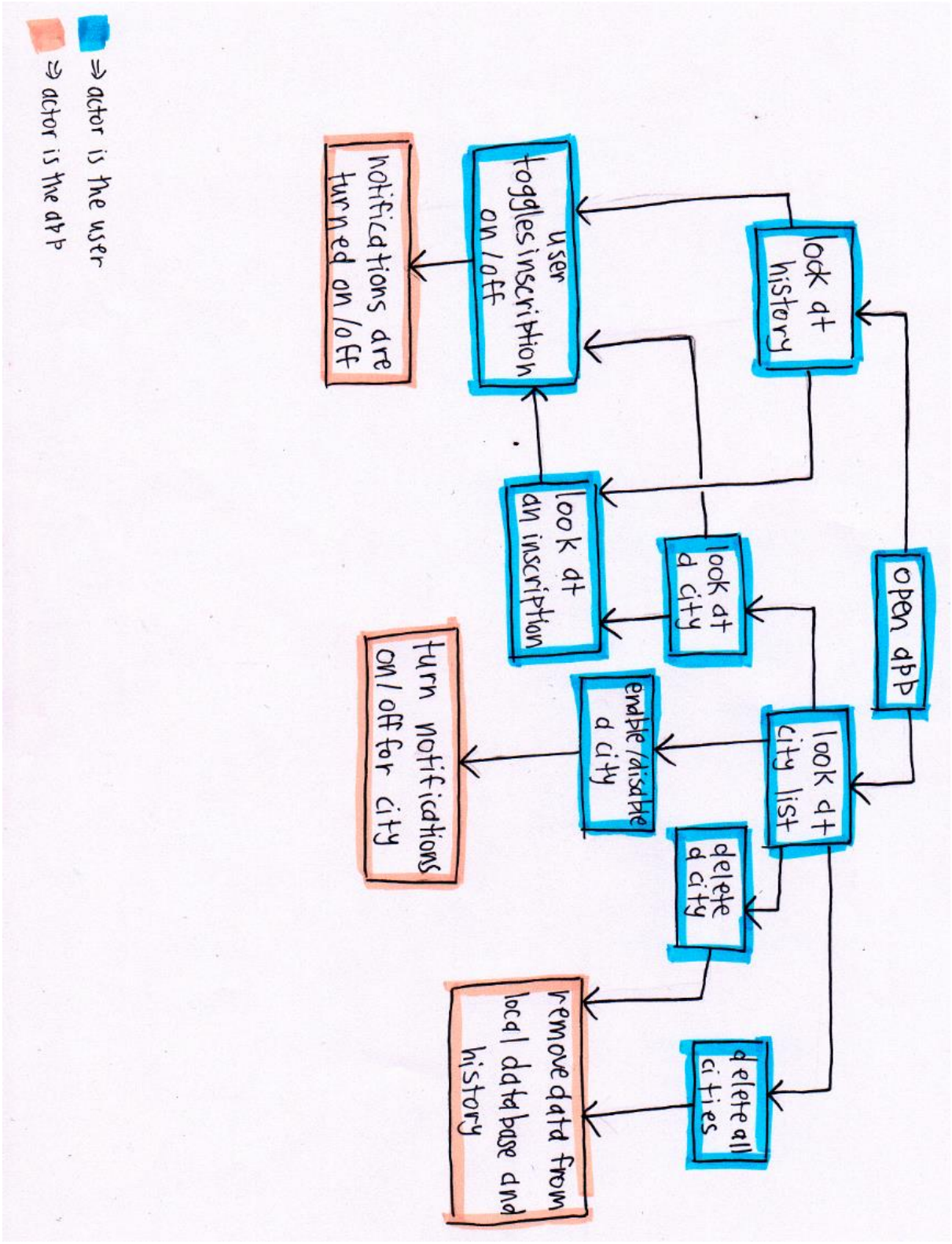


Figure 2. Workflows for when the app is opened

Figure 1 workflows:

1) The user gets a notification from the app saying they have entered an inscription area. The user opens up the app to look at the inscription and may or may not toggle notifications on and off for that inscription.
    i. Use case: Inscription geofence is entered (user and app).

2) The user gets a notification from the app saying they have entered an inscription area. The user does not open up the phone to check the inscription.

3) The user enters a new city geofence area. The user then gets a notification from the app indicating the app is storing a large amount of data. The notification suggests the user should delete data from one or more cities. The user either ignores the notification or opens the app and deletes one or more cities.

4) The user enters a new city geofence area. The user then gets a notification from the app indicating the app is storing a large amount of data and that the city the user has just entered will not be downloaded onto the phone. The user either ignores the notification or opens the app and deletes one or more cities.

Figure 1 use cases:

1) Enter new city geofence area (user/app)
2) Download city database via Wi-Fi (app)
3) Enter geofence/inscription area (user/app)
4) Send notification to user (app)
5) Receive notification (user)
6) Toggle inscription notification on (user)
7) Toggle inscription notification off (app)
8) Look at inscription after notification (user)
9) Ignore inscription after notification (user)
10) Add inscription to history (app)

Figure 2 work flows:

1) User opens app to look at the history (i.e. recently viewed inscriptions). The user toggles notifications for various inscriptions on and off.
2) User opens app to look at the history (i.e. recently viewed inscriptions). The user looks at a specific inscription. Information for that inscription is displayed. The user can toggle the notifications for the inscription on and off.
3) User opens app to look at the list of cities. The user looks at a specific city, which displays the list of inscriptions seen in that city. The user toggles notifications for various inscriptions on and off.
4) User opens app to look at the list of cities. The user looks at a specific city, which displays the list of inscriptions seen in that city. The user looks at a specific inscription.

Information for that inscription is displayed. The user can toggle the notifications for the inscription on and off.

5) User opens app to look at the list of cities. The user enables or disables notifications for one or more cities.
6) User opens app to look at the list of cities. The user deletes a city from the app's database.
7) User opens app to look at the list of cities. The user deletes all cities from the app's database.

Figure 2 use cases:

1) Open app (user)
2) Look at history (user)
3) Look at list of cities (user)
4) Look at a specific city (user)
5) Look at a specific inscription (user)
6) Toggle inscription notification on/off (user)
7) Turn notifications for an inscription on/off (app)
8) Toggle city notification on/off (user)
9) Turn notifications for a city on/off (app)
10) Delete one or more cities (user)
11) Remove one or more city databases from app's memory (app)

+ ---------- +
+ ---------- +

App Design

We have a very tentative and basic idea of what we envision our app's GUI to look like. We have drawn a rough image – including which objects of the GUI link to which other objects – and have added it to this document:

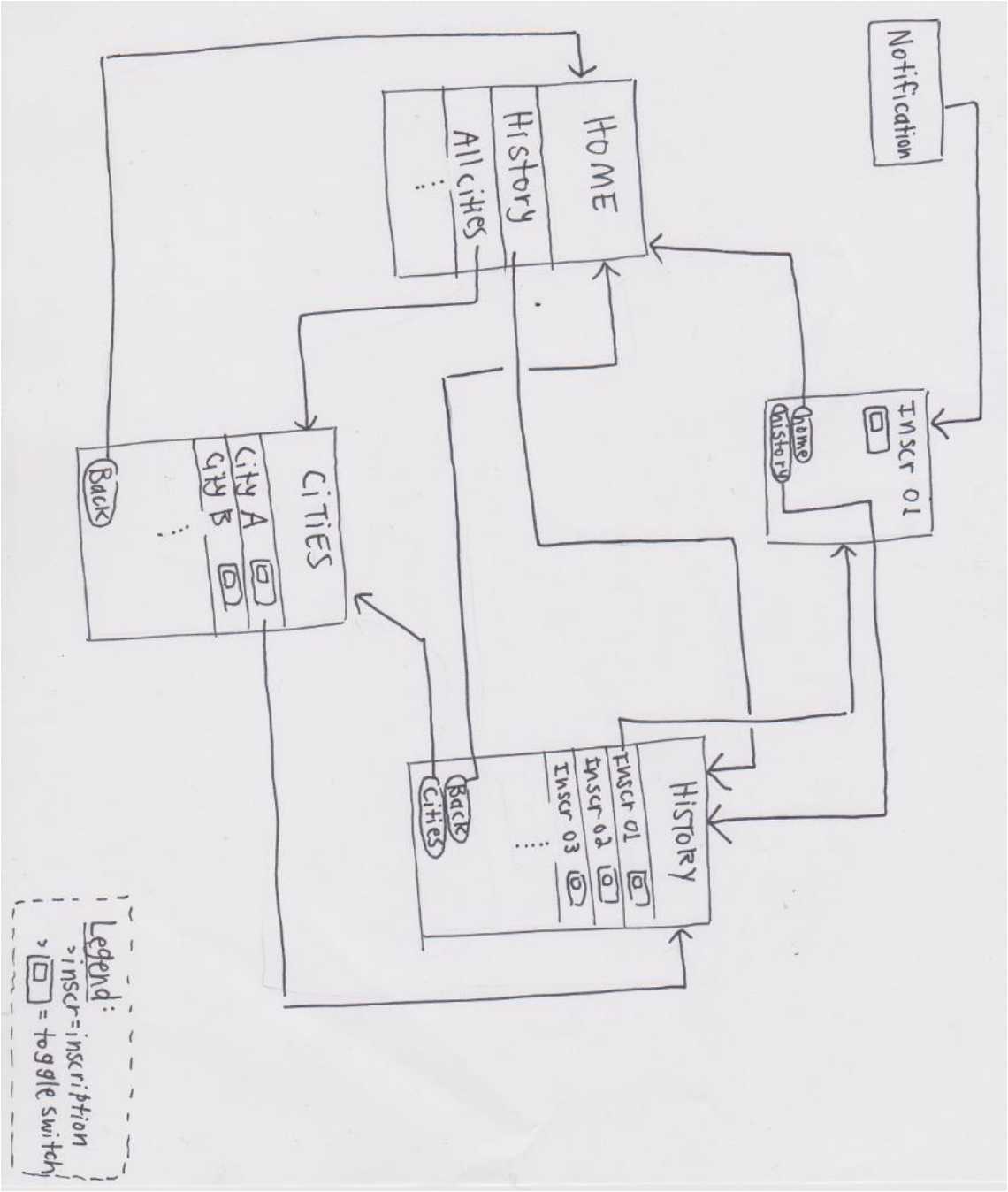


Figure 3. Potential GUI design