Project Description

The project we are doing is called ExploreKU which is a mobile application that can provide users information about the KU main campus. The idea was first proposed in EECS 448 as a course project. During that time we built and Android application that held the information such as buildings' info, addresses, departments, bus stops, and parking lots. We believe this project can provide KU students, especially freshmen, useful information about our campus and all the utilities available to them. For example, one can use this app to search for parking lots and bus stops around him/her. He/she can also use this app to understand and appreciate the stories behind each building on our beloved campus.

This capstone project is an extension toward that idea. In this project, we will introduce a new functionality called “Augmented Reality View”. The goal is to display aforementioned point-of-interest information as stylish overlays on the user’s camera view.

In addition to extending the functionalities, we are going to build a web service to host and respond to request to the location information previous stored in a embedded database. The mobile application will post a request to the server with its geospatial data, and the server will respond with the data about the locations around the user. This will not only improve the performance of the application, but also give great extensibility for future functionalities.
Project Design

As a mobile application that connects with a web server, this project will be composed of two main parts, the mobile application and the web application. The mobile application part has been developed in Android platform in our EECS 448 - Software Engineering course that we all have taken. However, this application incorporated a local database and lacked some functionalities that we intended to implement at that time, such as the Augmented Reality functionality and updatable database. From these reasons, we are planning to expand and develop the functionalities of this system more by implementing the functionalities mentioned.

In developing the web service system, we will be using Heroku as our deployment platform. Heroku is a Cloud Computing service that provides Platform as a Service (PaaS). This service supports many programming languages, is easy to deploy, and also very scalable. Also, using this third party platform, the services reliability of the web server can be guaranteed rather than using our personal PC devices as the web server. These characteristics makes this a perfect platform for our project. Hence, we will be developing, testing, and deploying our project in this platform.

As for the programming language that we will use for the development, we chose Ruby on Rails for our web application, mainly because Rails provides great productivity and extensive libraries (gems.) Ruby on Rails follows MVC (Model-View-Controller) patterns. The following picture explains how MVC works:
- Models are Ruby classes. They maintain the relationship between the objects and the database and handles validation, association, transactions, and more.
- Controller is the facility within the application that directs traffic, on the one hand, querying the models for specific data, and on the other hand, organizing that data (searching, sorting, messaging it) into a form that fits the needs of a given view.
- View is a presentation of data in a particular format (HTML, CSS, XML, Javascript, JSON), triggered by a controller's decision to present the data.

The web application will have 4 components: a database, a management dashboard, an authentication system, and an API service. First, we have a PostgreSQL database to store all the location data on KU main campus. Currently, we plan to have 4 tables:

- Building: includes the following fields: picture, history, department, amenities
- Bus stop: includes the following fields: bus number, bus schedule
- Parking lot: includes the following fields: permit color, operation hours
- Campus objects: blue phones, vending machines, ATM, etc.

Each of these entity will also have geo-spatial data embedded by using PostGIS (a software program that adds support for geographic objects to the PostgreSQL object-relational database.)
We will also create a management dashboard to manage the data. In specific, basic CRUD actions for each of the models will be implemented:

- Create or add new entries
- Read, retrieve, search, or view existing entries
- Update or edit existing entries
- Delete/deactivate/remove existing entries

In addition, we will implement a basic authentication system that only allows users with admin right to modify the data.

The last component is an API service to communicate with the mobile app client. The general process is that the client will send out a data request to the server on occurrence of events. We proposed using moving distance as a metric to trigger the event (e.g. if the user has moved 10 meters from the last location that sends the data request, then fires a new data request.) The request will be a simple GET request that has the user’s current GPS coordinates as parameters. Upon receiving the request, the server will get the user’s latitude and longitude and run a location data aggregation service. This service will query the database and aggregate all the data points within 10-meter radius around the user. Finally, the server format the information as JSON and send the response back to the mobile client. The mobile client will now parse the response and render the data points on the camera view.

This diagram shows the flow of the web service:
The mobile application will be implemented using Unity 5 for the sake of 3D graphic rendering and cross-platform deployment. The user interface will be a combination of user camera stream as the base view, 3D graphics rendered as overlays and Unity Canvas for top user interface. Unity has built-in APIs to read camera input streams as constantly updated textures and to access a smartphone's gyroscope and digital compass. 3D images can be rendered by layers by manipulating Unity’s rendering pipelines. Thus, we can use the phone’s rotation status to rotate the camera around. If there is a marker sprite in the camera’s view frustum, it will be displayed on top the camera image. On the right is a screenshot of the proof-of-concept application of the aforementioned method.
The above graph demonstrates the internal structure of the application. The top user interface will dispatch user actions to the data processing framework. If a data request is required, the data processing framework will dispatch a request to the network communication interface. The living singleton that implements the interface would carry the job of fetching the data. For testing, there would be an adapter that returns hard-coded data. For actual usage, there would be a framework to do HTTP communication with the web service using REST API and Unity’s WWWRequests. When the WWWRequests or adapter return, the data processing framework would receive and process the returned JSON string with callback function invoked by the network interface into data objects and pass it back to the UI controllers and map marker overlay controller to visualize them.

**Project Milestones**

1. Gather data that is needed for the application. It is mainly geographic coordinates and description of points in interest in Lawrence, KS. We will first try gather it from the officials, e.g. Lawrence Transit, KU Parking, etc.. If we could not get data from them, we would build all the info database manually. Proposed deadline: by the end of the fall semester.
2. Design the API specifications. Since we aim to implement the web service to use REST API with, we need to specify the parameters and return values. Only by this we could focus our goal to implement the service and the connection manager of the AR application. Proposed deadline: by the end of the fall semester.

3. Set up and implement the web service. This includes handles all the query API aforementioned and building the database. Unit test is necessary for the consistency and should also be done by the deadline. Proposed deadline: before spring break starts.

4. Implement the AR application in Unity. This will include designing the user interface, system mechanics, and connection manager to the server. Testing with mock-up data should also be done by the deadline. Proposed deadline: before spring break starts.


6. Produce the documentation about the application, including the functions, classes, server, API, and every technical details used during the implementation process. Proposed deadline: before the due of the senior design project

**Ethical and Intellectual Property Issues**

1. Privacy issue

   For the privacy issue, we firstly mention the privacy of the location data in our map, which is sensitive. This location data refers to both the location data of the user and the location data regarding the point of interests in KU. Our server needs the location data of the users to return result correctly for ensuring our mapping information’s responsibility. Retrieving the location data of the user means that it is possible for outside party to sniff on the traffic created between the users’ mobile devices and the web server. This is very dangerous as the outside party could use the data for bad intentions to the users such as robbery or even as simple as stalking
their private lives, which we are sure are not somethings that our users would want to happen. Therefore, We'd better to protect the privacy of the sensitivity of the location data. Thus, encryption of transmitting location data will be necessary. To protect the location data of the users, we implement an encryption algorithm that will be run everytime the users’ mobile devices are communicating with the server. It can help us prevent the privacy of location data to be used for other ways and unintended reasons.

Another aspect that we need to pay intention in is regarding the storage of the users’ location data. We believe that not all user is comfortable with their location data being stored in our server. To assure the users that we do not have any hidden intentions regarding the user's' location data, we will not be storing any location data exchanged between the user's’ mobile device and the web server. This way, the users do not have to worry about their privacy regarding their location data. All of these represent our respect to the privacy and safety of important data.

2. Intellectual property

For the Intellectual property, we will use several libraries in the development of our application. As a result of that, in the process of our development, we should include the the license of each library which is used in our application somewhere. This way is not only show our respect to the intellectual property of these libraries, but also follow a good standard for the software development. The precondition of earning respect is showing sincerely respect to others. If anyone breaks the rule of intellectual property and misuses the product of other developers, it could make the hard-working developers earn nothing from their product. Actually, this action of violating the intellectual property will not receive severe punishment, so some people do not pay attention to this violation. That’s the reason we try our best to comply with the standard of intellectual property. On the one hand, it is the symbol that we’re programmers with excellent personal quality. On the other hand, we hope the users of our application could also be nice and honest to the intellectual property of our application.

3. Using the public data
The final part is about using the public data, the data we used are provided by the University of Kansas, which is a government agent. Therefore, any violations of privacy and intellectual property about these data will be seemed as a crime. Also, we have to pay attention to protect the public data in our storage without misuse from the outside. In order to prevent the violations to our public data, we need to consult with some responsible person who know profoundly about this issue to get the appropriate instructions on proper use of the public data. By communicating with the people related to security issues, we will know how to use the public data from KU properly without any violations, also about protecting the public data carefully to the invasion outside. It is not only responsible for the safety of our application, but also for the support and trust from KU. With the support and understanding of KU’s officers, we are just able to finish our application smoothly. Thus, we should make great effort in protecting their public data that we use.

Project Budget
We will host our web service on the cloud. To simplify our deployment process, we will use Platform-as-a-Service (PaaS) such as Heroku. This is our specific budget:

- **Hosting:**
  - Cost: $25 per container/month
  - Vendor: Heroku
  - Time: at deployment (April/May 2017)

- **Software licences:**
  - Cost: Varies
  - Vendor: Unity (plugins + assets), third-party programs and libraries
  - Time: as needed

- **Unity subscription:**
  - Cost: $35 per seat/month (Plus plan)
  - Vendor: Unity
  - Time: Throughout the development process
Work Plan

- Haonan Li: AR application design and implementation (priority), API specification, project management.
- Hao Luo: AR application implementation (priority), testing.
- Yang Zhou: API specification (priority), database design and collection, testing.
- Mohammad Isyroqi Fathan: web service implementation (priority), web service management and support, testing
- Hung Nguyen: Administration system (priority), API endpoints, testing.

Github Address

https://github.com/heshuimu/ExploreKU-phase-2-Unity-Application

Change Log

Added the repository for Unity application and changed work plan.

Gantt Chart

Please see the attached picture on the next page. Original image can be seen at:
https://drive.google.com/file/d/0B-8G5s2O_31DVUR4OVVsdEZxaVk/view?usp=sharing