Team 14 Final Project Design

John Freeman, Dilesh Fernando, Andy Kunz, Diego Soliz, Stephen Wiss

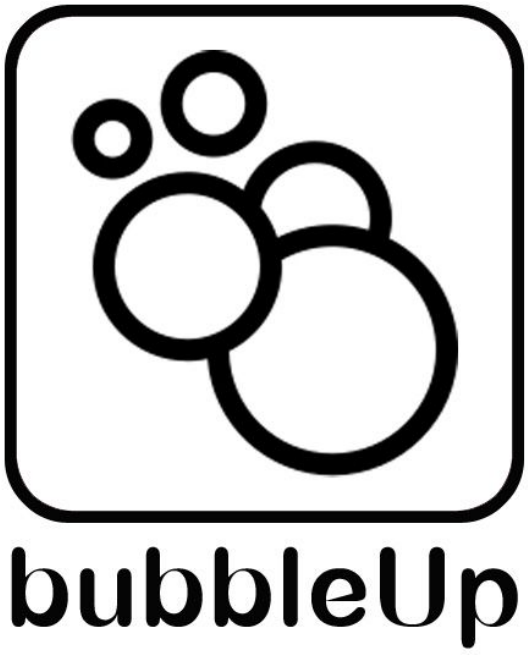


Table of Contents

General Team Information

Team Name: Team 14

Team Members and email addresses:

- John Freeman - j4freeman@gmail.com
- Andy Kunz - alkunz@ku.edu
- Dilesh Fernando - fernando.dilesh@gmail.com
- Diego Soliz - d481s306@ku.edu
- Stephen Wiss - s539w172@ku.edu

Team Meeting time: 11:45 Tuesdays

Lab Meeting time: 12:20 Tuesdays

Contact: John Freeman

Project Sponsor: None

Project Description

For our project, we will be creating an app that allows users to get up to date information about events and opinions of places around them. This will be handled through gathering data from the internet and a built in social media system allowing for users to directly share events, media, and opinions, while additionally be able to give feedback that will affect the visibility of other content. We will be using an Android frontend powered by a python based backend interfaced with a MySQL database via Heroku and NodeJS.

Users will be able to post pictures and events, and rate locations around them. This will be seamlessly tied into the backend which will scrape the web and social media to determine information about a certain area (given information is found regarding it). Events and pictures will be displayed as bubbles with extra information available if the user searches a location.

We intend to initially limit the scope of the application to Lawrence to prevent the need of large-scale processing and data storage assets as well as to facilitate speed of processing given the relatively small amount of computing resources we have available.

Project Milestones

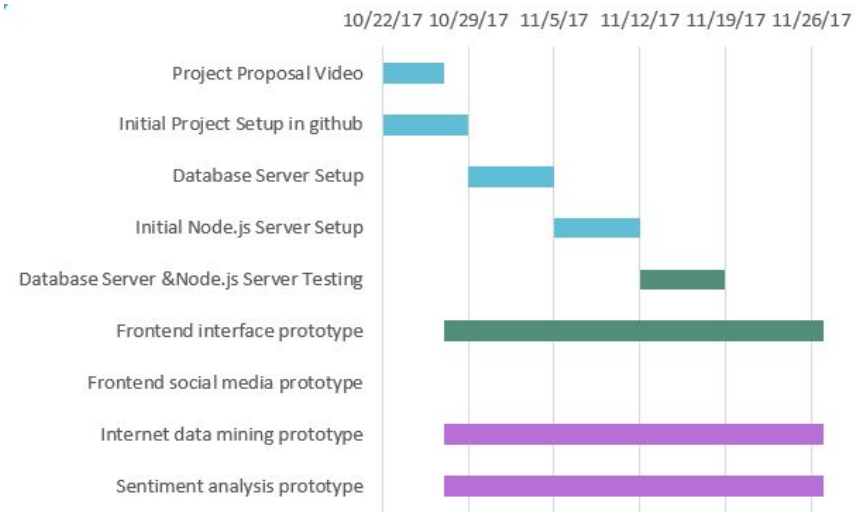
- **Fall Milestones**
  - Initial Documentation - 10/23/17
  - Initial Display containers for events on empty map related to location - 11/30/17
  - Initial Database Implementation: Create database tables and initial database structure - 11/12/17
  - RESTful API using NodeJS/PHP for data communication with database - 11/30/17
- **Spring Milestones**
  - Completed Data Analysis Per Location - 03/09/18
  - Completed Database System - 03/09/18
  - Completed Front End - 03/09/18
  - Full Integration of Database, Front/Backend - 04/09/18
  - Final Documentation of Functionality - 04/30/18

- **Gantt Chart Dates Table**

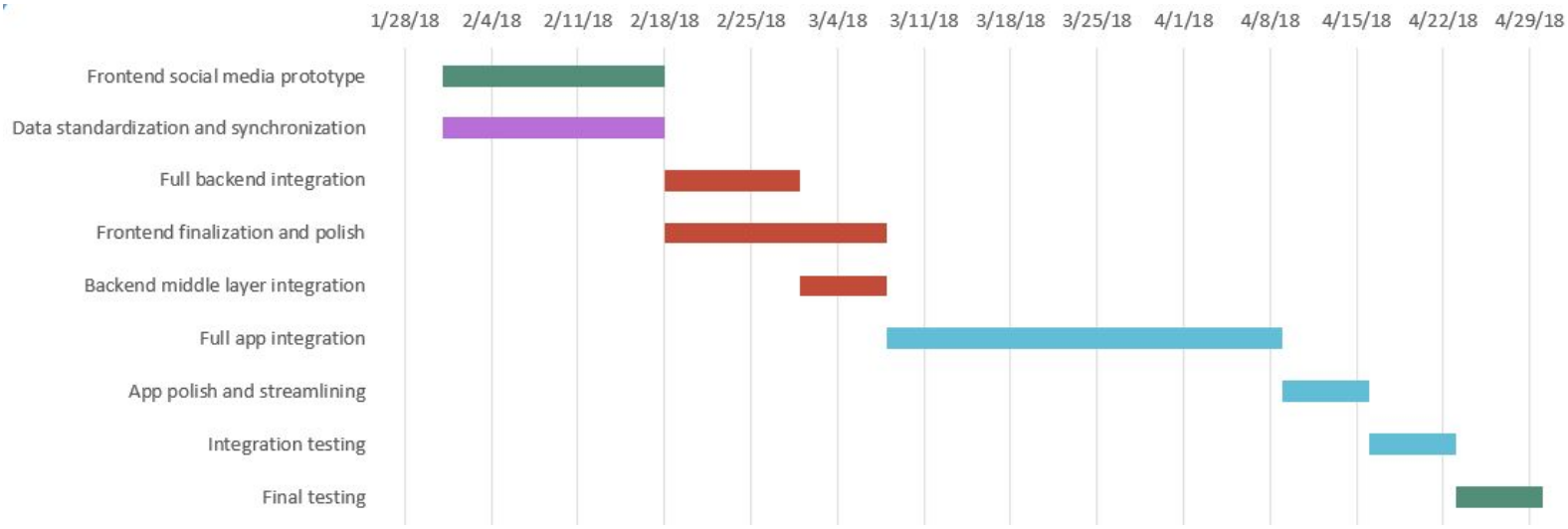
Task Name	Start Date	End Date	Duration
Project Proposal Video	10/22/2017	10/27/2017	5
Initial Project Setup in github	10/22/2017	10/29/2017	7
Database Server Setup	10/29/2017	11/5/2017	7
Initial Node.js Server Setup	11/5/2017	11/12/2017	7
Database Server &Node.js Server Testing	11/12/2017	11/19/2017	7
Frontend interface prototype	10/27/2017	11/27/2017	31
Frontend social media prototype	1/31/2018	2/18/2018	18
Internet data mining prototype	10/27/2017	11/27/2017	31
Sentiment analysis prototype	10/27/2017	11/27/2017	31
Data standardization and synchronization	1/31/2018	2/18/2018	18
Full backend integration	2/18/2018	3/1/2018	11
Frontend finalization and polish	2/18/2018	3/8/2018	18
Backend middle layer integration	3/1/2018	3/8/2018	7
Full app integration	3/8/2018	4/9/2018	32
App polish and streamlining	4/9/2018	4/16/2018	7
Integration testing	4/16/2018	4/23/2018	7
Final testing	4/23/2018	4/30/2018	7

- **Gantt Chart**

Fall Semester



Spring Semester



Project Budget

We do not intend to buy any special equipment for our project.

Work Plan

- John Freeman - backend
- Andy Kunz - frontend
- Dilesh Fernando - database/middleware

- Diego Soliz - frontend
- Stephen Wiss - backend

Github link

Our GitHub repository is located at: https://github.com/dangelox/bubbleup-api

Final Project Design

High-Level Design

At a high level, our application has three components: a front-end, middle layer, and backend. These components will individually be described in more detail later on. In terms of interaction, the backend and frontend will only communicate with the middle layer. The middle layer will route transactions to and from the database, so all communications can be standardized and allow for easy modification of our routines. We intend to have common access methods to facilitate ease of communication and access.

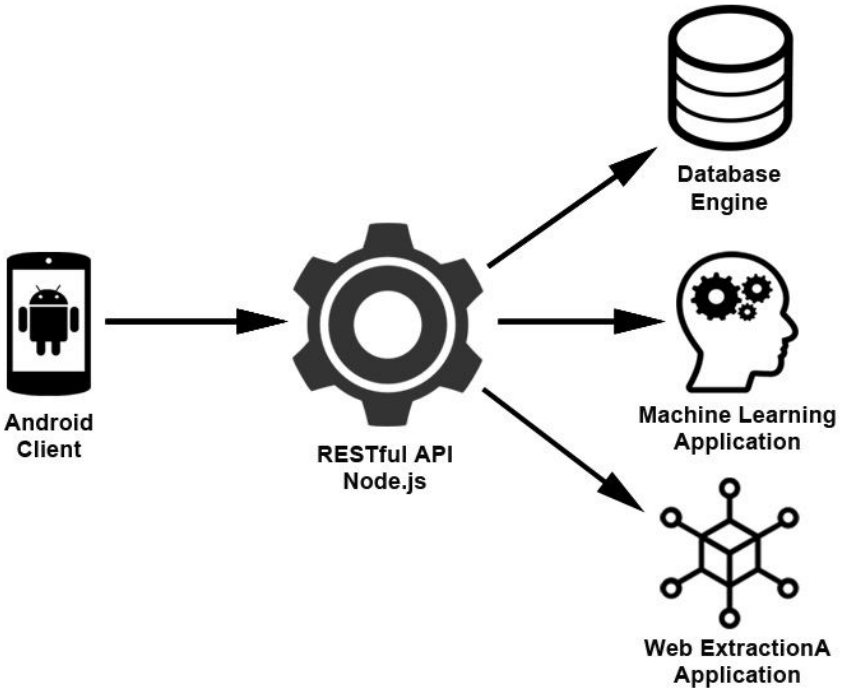


Figure 1. High-Level Design

Frontend - Android Client

The main interface of the application displays a google street map centered on the current user location, the application proceeds to fetch relevant information from a database. This information is displayed on screen in the form of bubbles (circles) hovering around the location where said information originated from or it is referring to. For example, if the retrieved data is a news article then its location on the map is determined by whether the source of the data is known or if the data itself has any mentions of a specific location.

Bubbles can vary in size and because of this so will their visibility on their map, popular bubbles will be visible from a zoomed-out map perspective. To avoid bubbles from overlapping others, upon zooming in on a location bigger will vanish if they are not originated from a location in the current map displayed, so as to make lesser bubbles visible.

To add clarity the bubbles will have a small icon indicator attached to them to let the user know what kind of information is contained within the bubble, the range of types of bubbles include but are not limited to: News Articles, Tweets, Activities, Events, User Posts, Images, etc.

Users are not required to create an account to be able to visualize data. The benefit to having an account is that users will be able to have a more customized feed based on their interests as well as be able to make posts and vote on which bubbles should have more visibility.

To display a quick overview about the contents of a bubble the user can tap on a bubble see a preview. Additionally a different method for browsing will sort the bubbles and make them a scrollable list to make their previews effortless to navigate through, so as to avoid the user from getting tired to tap many bubbles.

Bubbles will decrease in size as time goes on based on a decay variable we create and eventually disappear. This insures only the most relevant and current data stays on the map. If a story becomes very popular the bubble will get larger, working against our decay variable. This will allow for relevant/popular stories to stay visible on the map for longer amounts of time. As stories get older, their decay variable will increase to ensure that all stories eventually are taken off the map. This way a constant amount of views per week would not keep a story on the map indefinitely.

We also intend to make available some global information about the area the user is centered on, including net sentiment as well as common words/phrases being used or the most popular type of post.

A concept UI for the application is depicted on Figure 2.

Figure 2. Concept Design for UI (Subject to change)

Middle Layer - RESTful API

It is important that our Android mobile application is able to communicate with backend applications without depending on the underlying operating system or the programming languages. As such, we are implementing a Middle layer to our application so that our Android client can communicate with a suite of backend applications.

In modern mobile application development RESTful APIs are used to make applications distributed and independent over the internet with the aim of enhancing the performance, scalability, simplicity, portability, and reliability of the application. Its stateless software architecture provides robust model for client server communications. RESTful APIs are web services implemented using HTTP protocol allowing easy and standardized data communication.

We will be running Node.js as our middle layer RESTful API client that communicates messages to and from Android client to the backend services. Such as database connection, web extraction application, machine learning application. To implement this, we have several components, described in more detail below.

All communications to and from the client will use standard HTTP methods for backend communications, and will be in JSON format. The middle layer will handle processing and directing those objects to their proper formats and locations.

For user authentication, we are using a Node.js module named Password.js. Authenticated users will get an unique token from Node.js server and all communications to the middle layer should include the token in the HTTP header.

For our database connection, we intend to create several components: user authentication, user sign up, save user post, and post retrieval.

User authentication will verify the user exits in database. If the credentials match, the process is completed and the user is granted authorization for access. New user sign up will add new user information to the database. Save user post will save user posts and meta data (location, time, etc.) to database. Lastly, post retrieval will retrieve all the posts that are associated with a user's location and send to client.

When a user logs in, we will take the additional step of taking their GPS location and scraping social media for posts in their vicinity. This specific implementation is subject to change: we are unsure we will be able to do this fast enough to guarantee good load times and may have to move to a periodic sweeping model.

Backend - Database, Web Extraction, Sentiment Analysis, Data Engineering

Our backend will consist of several components: internet media mining, sentiment analysis, and data engineering.

For the database component, we are planning to use MySQL, an open-source relational database management system (RDBMS). Separate databases will be implemented to house data for backend applications, including: user account information, unsanitized data, displayable data, and others.

The user data will encompass all user information (name, login credentials, etc.) to facilitate logging in as well as maintaining preferences. The unsanitized data will contain unsanitized data acquired online that needs to be formatted to a displayable entry as well has have analysis performed on it. Unsanitized data will be periodically pulled, cleaned, and where applicable: analyzed. It will then be moved to a display database where it will then be accessible to users.

We are anticipating a lot of user activity stemming from logins and posts on the user and comment databases during peak usage time. Since the usage of these will likely be much more frequent than other data we believe storing them separately will increase load times and allow easier access of our data.

While our app will have native social media functionalities, we also intend to acquire data from the web for display within the app. We initially intend to limit this to a few news sources and twitter, but will expand our sources if we have time. To guarantee that our sources are from the lawrence area, we will only use tweets that are geolocated within the Lawrence area, and news from local stations and channels. Tweets will be located where they were tweeted, and will be subjected to sentiment analysis so we can get more information to the user in the bubbles. We intend to use RSS and the twitter API to gather this data and then engineer it to our format.

We will be gathering tweets for display on our application, and where relevant we intend to analyze the sentiment behind them, primarily through Python's Natural Language Toolkit. We hope to be able to implement this on a selective basis, thus only displaying sentiment of tweets where it makes sense and not calculating it otherwise. We are likewise considering implementing sentiment analysis for the native social media features, however we will likely

implement that such that only global sentiment will be changed, as we feel that classifying user posts would take away from the experience of that particular user.

We will need to process data on the backend, as we will be gathering it from a variety of sources and need to compress it down into common formats so it can easily be translated by the frontend. This will primarily be accomplished in Python, and the methodology involved in processing will vary depending on the data source. For the data from the native social media implementation little engineering will be required but for data acquired from twitter or news sources we likely will need to process the data from the APIs used into a common format for the database to store.

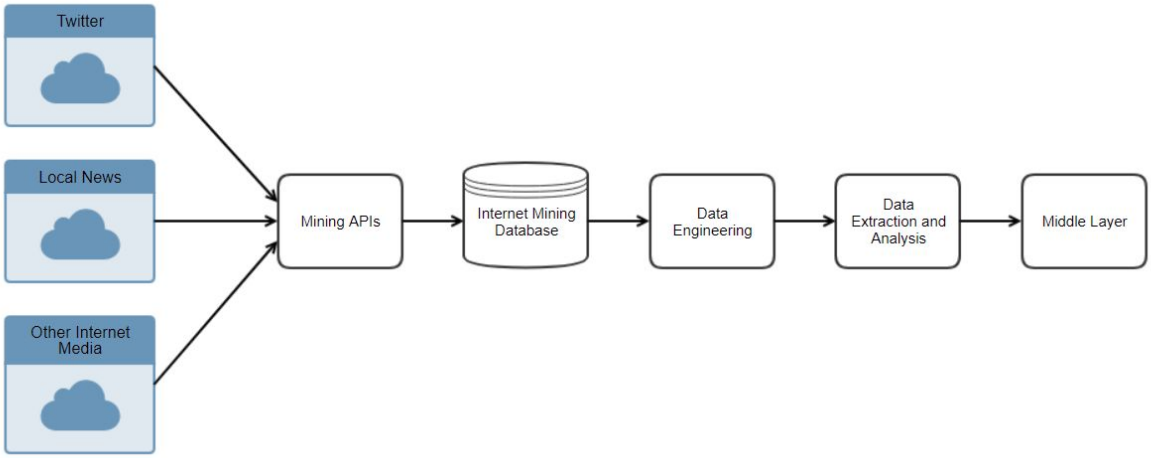


Figure 3. High Level Design of Backend

Ethical and Intellectual Property Issues

Ethical issues regarding our project will primarily stem from those of privacy. Our application will be inherently and purposefully geo-enabled, meaning that people's locations will be shared and that tweets will be shown based on proximity rather than content.

For user's of the native social media functionality, there are few concerns as the geolocation features are the reason they would be using the app. They will be made to accept that their location will be shared as part of signing up for the application, and the android operating system requires users to explicitly allow third-party applications to utilize location services, thus ensuring that the user is aware that their location and information is being shared and to only share when and what they want to.

For tweets that are gathered from twitter, we will need to only display tweets that are about public events rather than personal information, so as to protect privacy. Since we will be primarily determining tweets based on the location they were sent from, we will need a second layer of filtering to ensure that the tweets are meant for the public and are relevant to our user base. Our strategy for this is to create a filter powerful enough to only show tweets that are

public and relevant, but in the event that we are unable to do so we will add a functionality to the app to allow users to flag tweets as personal or irrelevant.

We do not anticipate any significant intellectual property issues with the project. The idea appears to be novel from our market research, so the primary IP issues will stem from frameworks and tools we use. We will carefully examine the licensing agreements of all tools we use and use and credit them in accordance with their agreements.

The project itself will be the sole property of team 14, as it is classified as a Student Academic Creation in Section A Part 5 of the Intellectual Property Policy for the Lawrence Campus document by the University.

Change Log

We initially planned to buy some unity assets to make our frontend map functionality easier, however when we looked into that with more detail we found that existing packages are too limiting in backend connection features and so will build our own over Google's Maps API.

We planned on having machine learning be a more major part of the project but realized that there were not sufficient packages and utilities to make it feasible within the development timeframe.We have as such downgraded the scope of that to encompass only simple sentiment analysis as well as basic summarization.

Initially we planned on having several separate databases to house information, and while we may still need to do that the document was adjusted to be more specific to current plans.