**GeoPost**

Michael Davis [davism78@uw.edu](mailto:davism78@uw.edu)

Megan Drasnin [mdrasnin@uw.edu](mailto:mdrasnin@uw.edu)

Ethan Goldman-Kirst [egk35@uw.edu](mailto:egk35@uw.edu)

Matthew Hertogs [mhertogs@uw.edu](mailto:mhertogs@uw.edu)

Neil Hinnant [nrhinnan@uw.edu](mailto:nrhinnan@uw.edu)

Katherine Madonna [madonk@uw.edu](mailto:madonk@uw.edu)

Andrew Repp [ajrepp@uw.edu](mailto:ajrepp@uw.edu)

Duncan Smith [duncan26@uw.edu](mailto:duncan26@uw.edu)

**Software Toolset**

Since GeoPost is an Android app, all developers will be using the Android SDK extensively. This will require Java and XML programming.

One of the central features of GeoPost is the map activity, which allows users to view message locations and their own current location. We will use the Google Maps API to add the map to our application, and we will use Google’s Location API to make our app location-aware. The Google Maps API is a good choice because most users are familiar with Google Maps and because the API and key are freely available. Google’s Location API will work well for us because it is simple, power-efficient, and gives immediate access to the user’s most recent location.

To store data associated with the app, including messages, their associated location coordinates, and user data, we will use Facebook’s Parse platform. Parse is a database technology stack that features strong integration with Facebook and a well-developed location-based posting system (Parse GeoPoint objects). The database is optimized for location-based queries, allowing us to, for example, get all points within a certain area in one Parse query. Parse simplifies our application significantly because it is already integrated with Facebook and has been engineered for use with location-based applications like GeoPost. Parse has several proof-of-concept code examples that demonstrate similar functionality to that of GeoPost. Additionally, unlike AWS, Parse has a free tier of service that does not charge for overages. Parse will serve up to 30 requests per second without cost, and will drop any requests over that threshold. It requires no credit card, and it is impossible for us to incur unwanted charges.

A key feature for our app is storing user data. This feature would require some form of authentication. To accomplish this, we will use Facebook SDK for Android, integrated through the Parse framework. Facebook integration is the best choice for authentication because it is secure, usernames are guaranteed to be unique, and it would allow us to integrate friend-specific features into our app via Facebook's Graph API. Further, because GeoPost is a social app, Facebook is a natural choice for authentication.

We will use GitHub for version control and bug tracking. We decided to use GitHub because it has integrated bug tracking, allows for collaborative code reviews, and is one of the most popular code hosts today.

**Group Dynamics**

Our project manager will be Neil Hinnant.

Every member of our group will share in the development. Neil might do less coding than other group members since he has the added responsibility of project management. Duncan will be primarily responsible for black-box testing, and Katie will handle most of the UI design.

We will divide group members into pairs, and assign each pair one feature (or sub-feature) to implement. This will ensure that each group member will have at least one other teammate to work closely with throughout development. Partners will be responsible for reviewing each other’s code, since they will already be familiar with the APIs used in the code.

**Risk Summary**

One risk we face is that none of us have extensive experience with Android development. To reduce the impact of this, we are watching Android tutorials, and we are making simple Android apps to practice using the Android SDK.

Another risk that we face is developing and maintaining a comprehensive and efficient integration test suite. Although all team members have experience with unit testing, none of us has significant experience with systems or integration testing. Further, we do not currently know Android best practices for developing a testing framework. To combat this risk, we have assigned one member of our group, Duncan, the SDET role. He will have primary responsibility for the testing framework, and he will spend the majority of his time designing tests and the commit framework we will use to prevent bugs from entering the master branch of our git repository. We have decided to add Neil to the SDET team, because we feel there should always be more than one person per feature or team.

A third challenge that we face is coordination. There are eight people on our team, and it is difficult to schedule meetings for which everyone can be present. Because there is so little face-to-face time, the potential for miscommunication and ideological drift exists. We have several systems in place to help mitigate and even eliminate this problem: we use a mailing list for group communications, we document all work assignments clearly in group meeting documents, and we have access to each other’s personal contact information so that issues can be resolved in a quick and decisive fashion.

Another challenge we may face is supporting all chosen versions (4.0 and up) of Android. Our research has suggested that an app as simple as GeoPost will easily run on all versions of 4.x, so it is unlikely that supporting all chosen versions of Anroid will be problematic. If this problem does occur, we would have to decide between eliminating support for some versions and writing more code to support the different versions of 4.x.

A final risk we face is that the database may become more complicated than we anticipated. We know that our database in initial form will likely look much different than our ideal database with all stretch features implemented. A text-based database is much simpler in principle than a mixed-media database, or one storing data for a complex profile system. Even so, we believe that our goals for GeoPost are not particularly ambitious in terms of database complexity. Because we only expect to see this problem if we are implementing a stretch feature, our initial mitigation plan is simply to drop the feature. If we have a surplus development time, we can return to it later and fine a more efficient implementation

**Usability Feedback**

We will perform user tests at two points in development: once in preparation for the Beta release (around May 5th), and once after we have completed the release candidate (around May 29th). Feedback will be most useful at these times because they mark the completion of several major features (see the schedule below). For each test, we will ask at least five non-CSE majors to spend an hour each using GeoPost. We will document the users’ reactions to our app, paying particular attention to the aspects of the app that users found confusing. At the end of each user test, we will ask the user for general feedback on his or her experience using GeoPost.

## **Project Schedule**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Milestone**  **Code** | **Major Milestone** | **Sub Milestones** | **Due Date** | **Effort Estimate**  **(person days)** | **Implementation**  **Dependencies** | **Test Dependencies** |
| 1 | **Zero Feature Release** |  |  |  |  |  |
| 1a |  | Make basic Android app | 4/27 | .75 | None | None |
| 1b |  | Add basic Android activity to app | 4/28 | .375 | 1a | 1a |
| 1c |  | Determine baseline build process | 4/29 | 1.25 | 1a,1b | 1a,1b |
| 1d |  | Determine how to make app available | 4/29 | .75 | None | None |
| 1e |  | Write documentation for users | 4/30 | 2.25 | None | None |
| 1f |  | Write documentation for developers | 4/30 | 2.25 | None | None |
|  |  | **TOTAL** | 5/2 | 7.625 |  |  |
|  |  |  |  |  |  |  |
| 2 | **Main Map Page** |  |  |  |  |  |
| 2a |  | Display Google Map | 5/3 | .75 | 1 | 1 |
| 2b |  | Center map at user location | 5/3 | .75 | 1,2a | 1,2a |
| 2c |  | Display Post button | 5/4 | .125 | 1 | 1 |
| 2d |  | Swipe out to blank settings screen | 5/4 | N/A | 1 | 1 |
| 2e |  | Clickable markers on map (No content) | 5/5 | .25 | 1, 2a | 1,2a |
|  |  | **TOTAL** | 5/16 | 1.875 |  |  |
|  |  |  |  |  |  |  |
| 3 | **Database Integration** |  |  |  |  |  |
| 3a |  | Connect to database | 5/3 | 2.25 | None | None |
| 3b |  | Implement DBQuery and DBStore | 5/3 | 3 | 3a | 3a |
| 3c |  | Create database interface (setup Parse) | 5/4 | .625 | 3a,3b | 3a,3b |
| 3d |  | Allow querying/updating from Android app | 5/5 | .625 | 1,3a,3b,3c | 1,3a,3b,3c,3d |
|  |  | **TOTAL** | 5/16 | 6.5 |  |  |
|  |  |  |  |  |  |  |
| 4 | **Dropping Pins** |  |  |  |  |  |
| 4a |  | Display pins with content from database | 5/9 | 1.5 | 1,2,3 | 1,2,3 |
| 4b |  | Add a pin with content to database | 5/10 | 1.5 | 1,2,3 | 1,2,3 |
|  |  | **TOTAL** | 5/16 | 3 |  |  |
|  |  |  |  |  |  |  |
| 5 | **Facebook Integration** |  |  |  |  |  |
| 5a |  | Register app with Facebook | 5/9 | 2.25 | 1 | 1 |
| 5b |  | Create login page with Facebook login button | 5/10 | 9 | 1,5a | 1,5a |
|  |  | **TOTAL** | 5/16 | 11.25 |  |  |
|  |  |  |  |  |  |  |
| 6 | **BETA RELEASE** |  |  |  |  |  |
| 6a |  | Ensure documentation is up to date | 5/14 | 1.5 | 1,2,3,4,5 | None |
| 6b |  | Write design changes and rationale | 5/14 | .25 | 1,2,3,4,5 | None |
| 6c |  | Prepare demo | 5/15 | .125 | 1,2,3,4,5 | None |
| 6d |  | Obtain customer feedback | 5/15 | .25 | 1,2,3,4,5 | None |
|  |  | **TOTAL** | 5/16 | 2.125 |  |  |
|  |  |  |  |  |  |  |
| 7 | **Settings** |  |  |  |  |  |
| 7a |  | Populate buttons | 5/18 | .25 | 2 | 2 |
| 7b |  | Display legend | 5/18 | .375 | 2 | 2 |
|  |  | **TOTAL** | 5/19 | .625 |  |  |
|  |  |  |  |  |  |  |
| 8 | **Profile Page** |  |  |  |  |  |
| 8a |  | Retrieve and display user pin count | 5/20 | .5 | 1,2,3,4,7 | 1,2,3,4,7 |
| 8b |  | Retrieve and display user viewed count | 5/20 | .5 | 1,2,3,4,7 | 1,2,3,4,7 |
| 8c |  | Retrieve and display  user information from Facebook | 5/20 | .75 | 1,2,3,4,7 | 1,2,3,4,7 |
|  |  | **TOTAL** | 5/21 | 1.75 |  |  |
|  |  |  |  |  |  |  |
| 9 | **FEATURE -COMPLETE RELEASE** |  |  |  |  |  |
| 9a |  | Obtain customer feedback | 5/22 | 2 | 1-8 | None |
| 9b |  | Ensure documentation is up to date | 5/22 | 1.25 | 1-8 | None |
| 9c |  | Test instructions for running | 5/22 | .625 | 1-8 | None |
| 9d |  | Set up automated building | 5/23 | 2 |  |  |
| 9e |  | Stretch feature hours | 5/23 | 4 |  |  |
|  |  |  |  | 9.875 |  |  |
|  |  | **TOTAL** | 5/23 |  |  |  |
|  |  |  |  |  |  |  |
| 10 | **RELEASE CANDIDATE** |  |  |  |  |  |
| 10a |  | Perform and document code reviews | 5/27 | 1.375 | 1-9 | None |
| 10b |  | Perform and document user tests | 5/29 | 1.125 | 1-9 | None |
|  |  | **TOTAL** | 5/30 | 2.5 |  |  |
|  |  |  |  |  |  |  |
| 11 | **1.0 RELEASE** |  |  |  |  |  |
| 11a |  | Perform extensibility exercise | 6/1 | 1.25 | 1-10 | None |
| 11b |  | Develop demo and presentation | 6/1 | 1.25 | 1-10 | None |
| 11c |  | Write requirements and schedule postmortem | 6/2 | 1.25 | 1-10 | None |
| 11d |  | Write individual retrospectives | 6/3 | 2 | 1-10 | None |
| 11e |  | Write class evaluation | 6/3 | 1 | 1-10 | None |
|  |  | **TOTAL** | 6/4 | 6.75 | 1-10 | None |
|  |  |  |  | 54.875 |  |  |
|  |  | **GRAND TOTAL** |  |  |  |  |