Newmont Mining Corporation
Hot Intercepts Ticker

Lauren Aberle, Thomas Brown, Derek Witte

I. Abstract

As one of the world's largest gold companies, Newmont Mining Corporation has the resources to acquire or invest in smaller mining companies that report successful findings. The Hot Intercepts Ticker Android application provides a business tool to display relevant drillhole intercept and stock information in order to assist analysts in determining Newmont's best course of action. The application displays the intercepts and stock prices for their respective companies in a user-friendly way. It features the use of the Google Maps API to geographically depict the intercepts and the Yahoo Finance API to present stock information.

II. Introduction

Being a competitor in today's market requires the ability to make informed decisions by analyzing important data. For Newmont Mining Corporation, this means being able to access significant exploration results and drillhole intercepts that are received from internal mining projects, competitors, and junior mining company press results. With this data and some stock price information, Newmont can identify companies or projects that they may wish to acquire or invest in. Currently, this information is summarized and manually distributed approximately every two weeks. Staying up to date on the latest information will give Newmont a competitive advantage.

The Hot Intercepts Ticker Android application allows Newmont analysts to view these results on a real-time basis. All intercept, project, and company information is presented in an intuitive manner. The application retrieves stock information from the Yahoo Finance API (application programming interface) and utilizes Google Maps to display project locations.

III. Requirements

A. Functional Requirements

1. The program must display a list of drill intercepts as they arrive.
2. The program must show real-time stock quotes from companies publishing the intercepts.
3. The program should have a map feature for easy geographic intercept referencing.
4. The program should have a search function to narrow down results that are displayed.
5. The program must have a convenient user interface that allows users to easily access needed information. Data can be presented in a basic format, and then an option should

be available to retrieve detailed information (e.g. map, additional intercept information, or stock charts).

B. Non-Functional Requirements

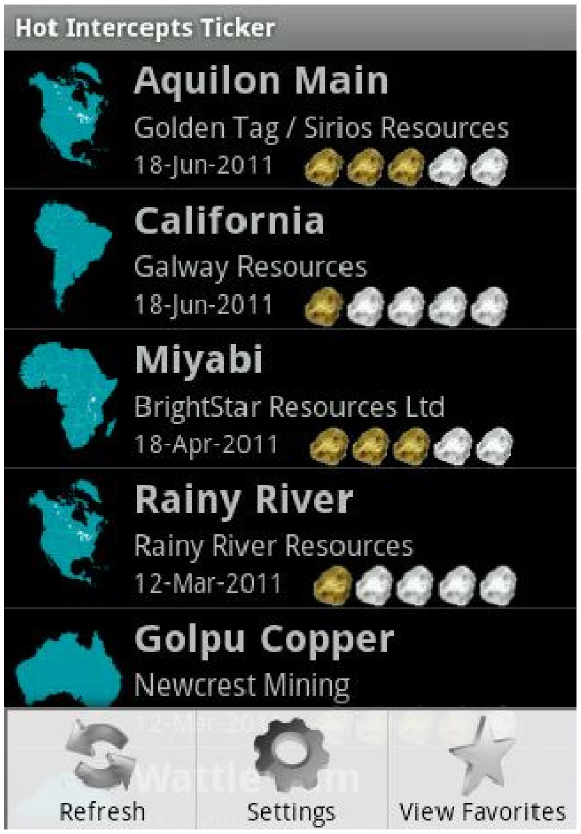
1. The program must use the Eclipse development environment.
2. The program must be written in the Java programming language.
3. The program must be an Android mobile application.
4. The program must utilize Newmont's Hot Drillholes database and the Intercepts Web Service.
5. The program must be uploaded to a Subversion repository on Newmont's servers.

IV. Design

A. User Interface Design

Our system is based upon the need for displaying geologic intercepts in a user-friendly way. The main screen, shown in Figure 1, displays a list of the most recent intercepts. Each intercept features the project name, company name, date received, and a ranking based on how much gold was found in the intercept. In the menu, there are options to refresh the page, view settings, and view favorites. The refresh button loads the most recent intercepts from the Newmont database. The settings button allows the user to select how many intercepts are displayed on each page, the number of intercepts stored in the local database, and the automatic refresh interval. It also provides an option to reset the database. To see intercepts a user has saved, he or she can click on the view favorites button. A search function is available through every Android's standard hardware buttons. It gives the user the option to find a specific string within the project and company names.

When the user touches an intercept on the intercept list screen, a new screen appears. Shown in Figure 2, the intercept details screen shows more detailed information about the intercept the user touched. The menu provides options for adding the intercept to a list of favorites and e-mailing the intercept information. There are also tabs to view company information, project information, and a map.


Hot Intercepts Ticker

Aquilon Main
Golden Tag / Sirios Resources
18-Jun-2011

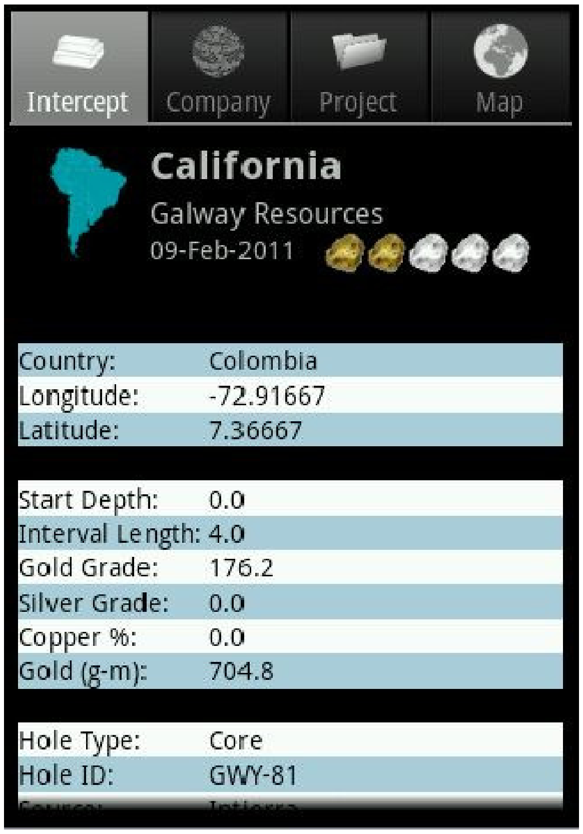
California
Galway Resources
18-Jun-2011

Miyabi
BrightStar Resources Ltd
18-Apr-2011

Rainy River
Rainy River Resources
12-Mar-2011

Golpu Copper
Newcrest Mining

Figure 1. Intercept List Screen



California
Galway Resources
09-Feb-2011

Country:	Colombia
Longitude:	-72.91667
Latitude:	7.36667

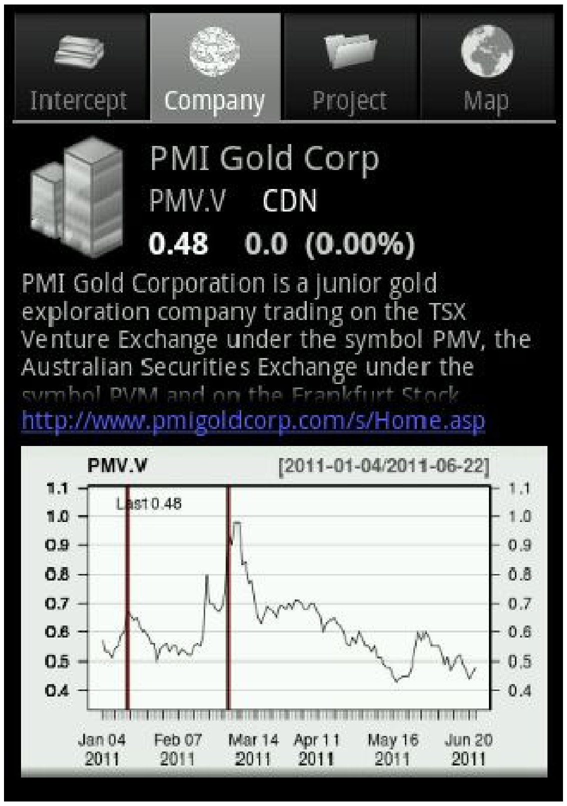
Start Depth:	0.0
Interval Length:	4.0
Gold Grade:	176.2
Silver Grade:	0.0
Copper %:	0.0
Gold (g-m):	704.8

Hole Type:	Core
Hole ID:	GWY-81

Figure 2. Intercept Details Screen

When the user touches the company tab, the company details screen appears, shown in Figure 3. This page features the company name, description, website, and detailed stock information from Yahoo. It also includes a chart of stock prices from the last 6 months. The chart has significant intercepts overlaid to assist in determining the role of intercepts in affecting a company's stock price.

The project screen, shown in Figure 4, features the project name, description, website and location, as well as other intercepts from that project. Touching an intercept displays the intercept details screen for that intercept.



PMI Gold Corp

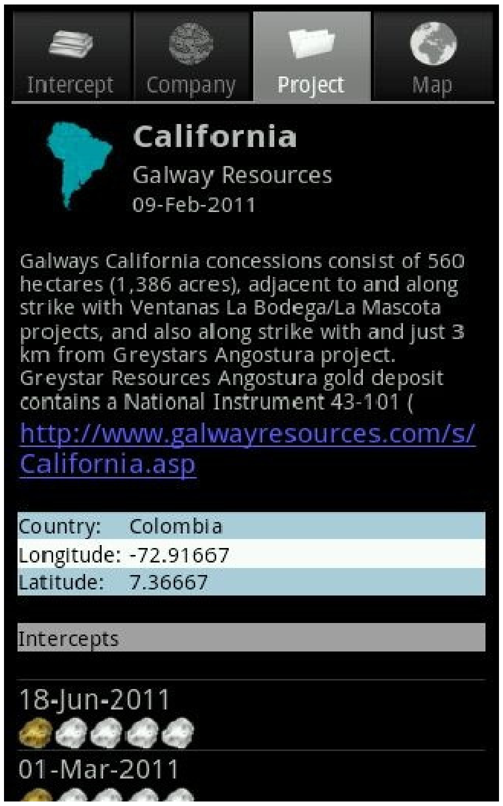
PMV.V CDN

0.48 0.0 (0.00%)

PMI Gold Corporation is a junior gold exploration company trading on the TSX Venture Exchange under the symbol PMV, the Australian Securities Exchange under the symbol PVM and on the Frankfurt Stock

http://www.pmigoldcorp.com/s/Home.asp

Figure 3. Company Screen



California

Galway Resources

09-Feb-2011

Galways California concessions consist of 560 hectares (1,386 acres), adjacent to and along strike with Ventanas La Bodega/La Mascota projects, and also along strike with and just 3 km from Greystars Angostura project. Greystar Resources Angostura gold deposit contains a National Instrument 43-101 (

http://www.galwayresources.com/s/California.asp

Country:	Colombia
Longitude:	-72.91667
Latitude:	7.36667

Intercepts

18-Jun-2011

01-Mar-2011

Figure 4. Project Screen

The map screen displays a Google Map with all projects in the area plotted on the map. Each project has an icon that represents the quality of the project's intercepts. This is shown in Figure 5.

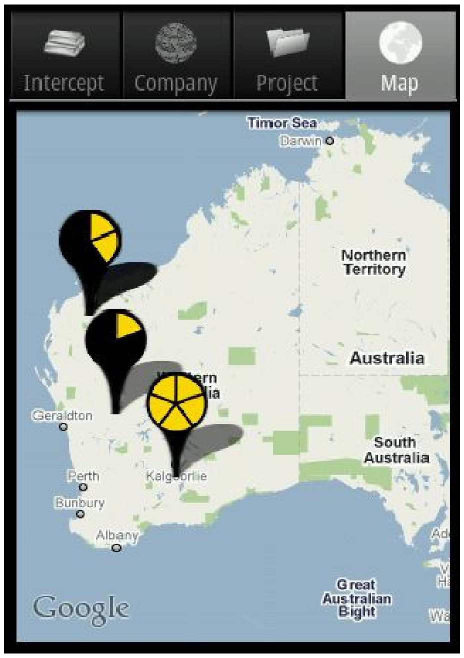


Figure 5. Map Screen

B. Program Design

Our application interacts with Newmont's Hot Drillholes Database through the Intercepts Web Service to retrieve intercept information. It also fetches stock information from the Yahoo Finance API. Figure 6 shows the architecture diagram for our system.

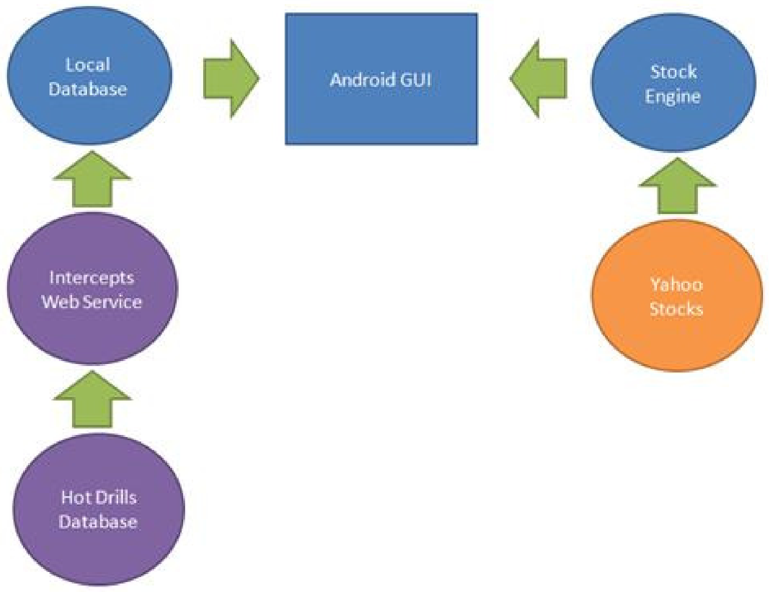
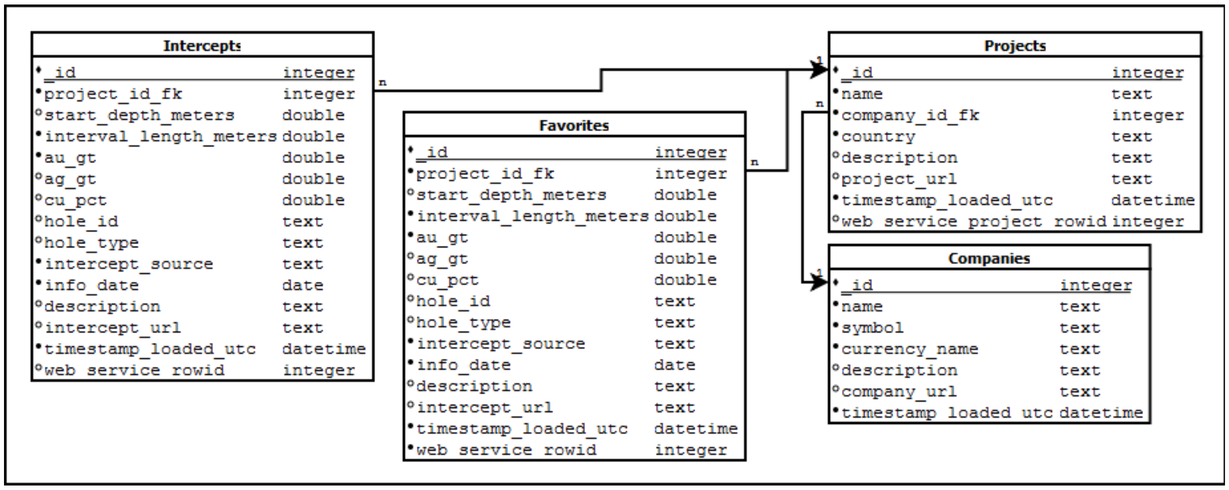


Figure 6. System Architecture Diagram

The stock engine gathers stock price information from Yahoo and sends it to the GUI to display. The intercept information is retrieved from the Hot Drills Database by the Intercepts Web Service. This data is stored into a local SQLite Database within the Android application. The graphical user interface (GUI) accesses this database to display the intercepts.

The application's local SQLite database allows for easier and faster intercept access. The database schema, shown in Figure 7, depicts the relationships between intercepts, projects, and companies and how they are stored in the local database. Each company can own multiple projects, and each project can have multiple intercepts. The schema is based on the actual structure of the Hot Drillholes Database to provide consistency, although there are a few extra fields in the local version to allow for application-specific features. For example, the web_service_row_id field in the intercepts table is used to prevent intercepts from being entered into the local database multiple times, allowing for efficient refreshing of the latest intercepts. Whenever an intercept is e-mailed, the web_service_project_row_id field in the projects table is used to generate the project website link. It directs the user to Newmont's web version of the project page, which was developed by our client concurrently in PHP (a web-based programming language). This gives the user access to even more information.



Intercepts	
• _id	integer
• project_id_fk	integer
∘ start_depth_meters	double
• interval_length_meters	double
• au_gt	double
∘ ag_gt	double
∘ cu_pct	double
∘ hole_id	text
∘ hole_type	text
• intercept_source	text
• info_date	date
∘ description	text
∘ intercept_url	text
• timestamp_loaded_utc	datetime
∘ web_service_rowid	integer

Favorites	
• _id	integer
• project_id_fk	integer
∘ start_depth_meters	double
• interval_length_meters	double
• au_gt	double
∘ ag_gt	double
∘ cu_pct	double
∘ hole_id	text
∘ hole_type	text
• intercept_source	text
• info_date	date
∘ description	text
∘ intercept_url	text
• timestamp_loaded_utc	datetime
• web_service_rowid	integer

Projects	
• _id	integer
• name	text
• company_id_fk	integer
• country	text
∘ description	text
∘ project_url	text
• timestamp_loaded_utc	datetime
∘ web_service_project_rowid	integer

Companies	
• _id	integer
• name	text
• symbol	text
• currency_name	text
∘ description	text
∘ company_url	text
• timestamp_loaded_utc	datetime

Figure 7. Database Schema

The application's internal structure is based around the screens in the User Interface Design section. Figure 8 depicts the application's class structure in UML. Each screen has its own Android Activity. The Map, Intercept, Company, and Project Activities are all contained within the DetailsTabHost, which allows for the use of tabs. In order to retrieve necessary information from external sources, the InterceptRetriever, Intercept, and InterceptXMLHandler classes gather and parse information from the Intercepts Web Service and the YahooStockRetriever, Stock, and StockXMLHandler classes fetch stock information from the Yahoo Finance API. The DatabaseManager class utilizes Android's SQLiteOpenHelper class to store information gathered from the retrievers into SQLite database tables within the application.

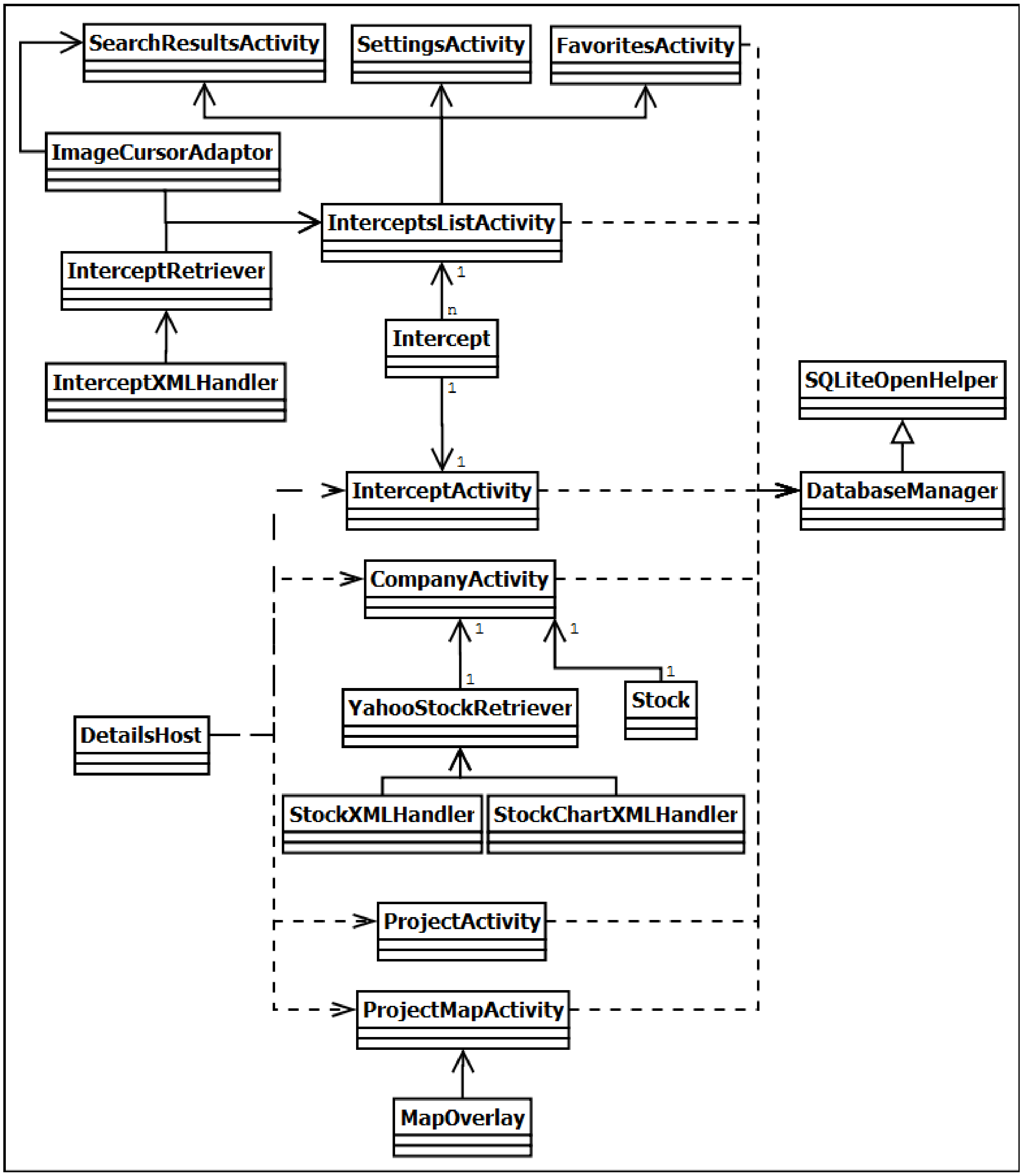


Figure 8. UML Diagram

V. Implementation Details and Results

A. Design Decisions

The decisions we made during the design of our application were crucial to the success of our project. The following are some significant choices we made.

We were offered the choice of developing either an Android application or an iPhone application. We selected Android because we have more experience with Java, we own Android devices to test the application on, and we wanted the ability to develop using the Eclipse IDE, which we were already familiar with. This decision required that we utilize Android's built-in SQLite database, which worked extremely well for our needs.

There are two widely used XML parsers used in Java - Simple API for XML (SAX) and Document Object Model (DOM). Although it required more coding, we implemented the SAX parser because it uses less memory and runs faster.

Our client preferred that we utilize the Yahoo Finance API to retrieve the stock information and chart displayed on the company screen. We initially retrieved the chart from the Yahoo Finance API, but overlaying intercepts on this chart was difficult and error-prone. Our client decided to develop another web service that uses a graphics development tool called R to generate the stock charts. These charts are more atheistically pleasing and allowed for overlaying intercepts easily. We ended up using this web service to retrieve the stock charts.

An optional feature our team chose to implement was a map view that allows the user to visually explore where intercepts are both geographically and relative to other intercepts. We decided to use the Google Maps API to implement this feature because it had extensive tutorials, the ability to overlay points, and readily available libraries.

B. Challenges

One of our first roadblocks involved becoming familiar with debugging in Android. Setting break points and using debug mode is roughly the same as any other Java project in Eclipse. However, when an Android application encounters a fatal error and crashes, the phone's screen doesn't show the error, and the stack trace is not printed to the console. However, an Android Eclipse tool, LogCat, displays the errors. LogCat can also be used in place of printing debug statements to the console. Once we found LogCat, debugging became much easier.

Another issue we discovered involving paginating the intercept list. In order to create a button that only appears at the bottom of the Android ListView (rather than always showing at the bottom of the screen), we had to create a class that extends Android's SimpleCursorAdapter to override the getView method. This method allows the programmer to specify the layout for each row in the ListView, which gave us the ability to make a list item that linked to the next page of intercepts.

Google requires an API key in order to use the Google Maps API. The key enables the mobile device to load map tiles from the Internet, and is dependent on the certificate used to sign the Android application. This was a small obstacle for our team as we have very little experience with API keys and security certificates. After searching the web extensively, we discovered how to obtain and use a Google Maps API key. Java includes a command line program named Keytool that can be used to find the MD5 fingerprint of the certificate, which in turn can be entered into Google's website to receive an API key. The API key is then entered into the XML file associated with the map. When in debug mode, all versions of the Android application are signed with a debug certificate; however, when the released version of the application is exported, it is signed with a different certificate. Different API keys must be used for each certificate used to sign the application.

C. Results

We put our final application through a rigorous set of acceptance tests to ensure that it functions in a way that is user-friendly. It incorporates all of the project requirements, successfully retrieving data from the required sources and presenting the intercept data in an intuitive way. It passed every test with flying colors.

VI. Conclusions and Future Directions

A. Future Concerns and Additions

Although we implemented all the required features and a few optional ones, there is always room for improvement. The following are some concerns and future additions we did not have the time or ability to address.

The version of the Hot Drillholes Database we used to test our application only had a small sample of intercepts (about 30). However, when the application is connected to the live database, there will be thousands of intercepts to download and display. Our client estimated that in the span of three years, there will be 5,000 to 8,000 intercepts in the database.

Unfortunately, we were unable to test our application with a larger data set. However, we believe that it should perform reasonably well, though there may be a few minor performance issues.

A problem may emerge if an intercept is deleted from the Hot Drillholes Database. Because our client didn't anticipate that many intercepts would be deleted and we had a limited amount of time to implement the application, we did not include a method to remove deleted intercepts. This issue would not cause any bugs, but it would not reflect the actual status of the database. Using the database reset button in the settings screen provides a solution, though it requires more work for the user, which is less than desirable.

Although we had hoped to write a few JUnit tests to cover the important sections of our code, we were unable to implement any automated testing methods within our time constraint. A future release may include this testing, which focuses on testing database interactions using EasyMock. EasyMock is an open source software that provides mock objects and methods convenient for testing, providing testing ability for portions of code that modify production databases without altering the actual database.

B. Conclusions

Our application addresses Newmont's basic needs for a mobile application that organizes and presents drillhole intercept data in real-time. We provided a simple user interface that allows the user to easily view general information about many intercepts, or specific information about one intercept. The detailed screen views include intercept details, company stock information, general project information, and a map. We were also able to include a number of optional features such as the ability to store favorites and e-mail intercept information.

VII. Appendices

The Activity Diagrams show how the user interactions work. Figure 9 shows what occurs when the user starts the application (a list of intercepts is downloaded and displayed). After the intercept list displays, the user can perform a variety of actions from the menu (shown in Figures 10, 11, 12, and 13). Figure 10 depicts the functionality of the favorites list; Figure 11 demonstrates how the refresh button works; Figure 12 exhibits the functionality of the search feature; Figure 13 displays how the settings feature behaves. Notice that Figure 10 has a branch that adjoins with Figure 14.

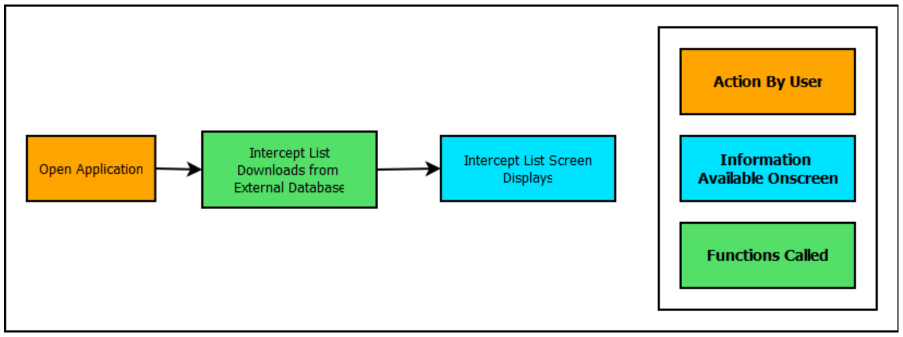

Open Application
Intercept List Downloads from External Database
Intercept List Screen Displays
Action By User
Information Available Onscreen
Functions Called

Figure 9. Application Start-up Activity Diagram

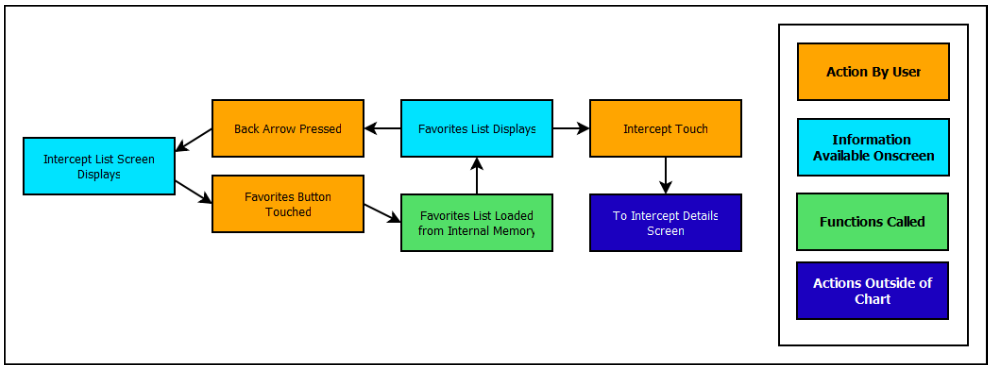


Figure 10. Favorites Activity Diagram

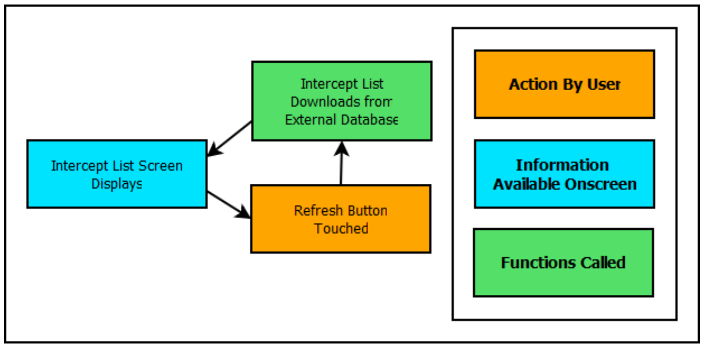

Intercept List Downloads from External Database
Intercept List Screen Displays
Refresh Button Touched
Action By User
Information Available Onscreen
Functions Called

Figure 11. Refresh Button Activity Diagram

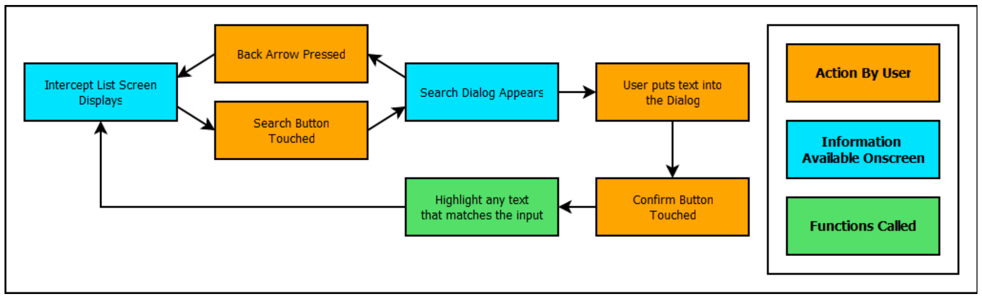

Intercept List Screen Displays
Back Arrow Pressed
Search Button Touched
Search Dialog Appears
User puts text into the Dialog
Confirm Button Touched
Highlight any text that matches the input
Action By User
Information Available Onscreen
Functions Called

Figure 12. Search Button Activity Diagram

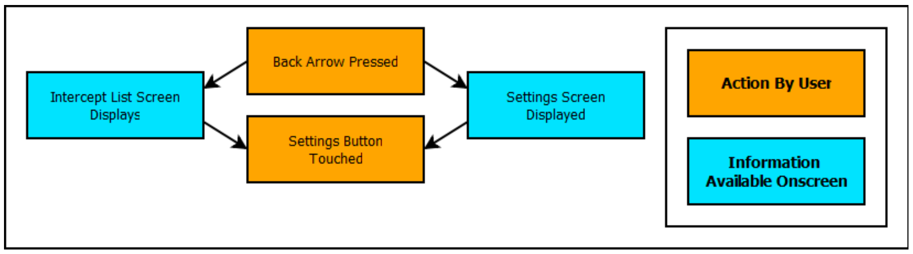

Back Arrow Pressed
Intercept List Screen Displays
Settings Screen Displayed
Settings Button Touched
Action By User
Information Available Onscreen

Figure 13. Settings Activity Diagram

Figure 14 shows what occurs after the user selects an intercept from either the intercept list or favorites screen. Each of the screens downloads the information from the internal database and displays it on screen. The intercept, company, and project screens include a website link, which, when clicked, opens the Android's web browser. The intercept screen is capable of sending an e-mail about the intercept. There is also the ability to add the current intercept to the user's favorites list. On the project screen, the user can see a list of intercepts belonging to the project, and can select one to view that specific intercept.

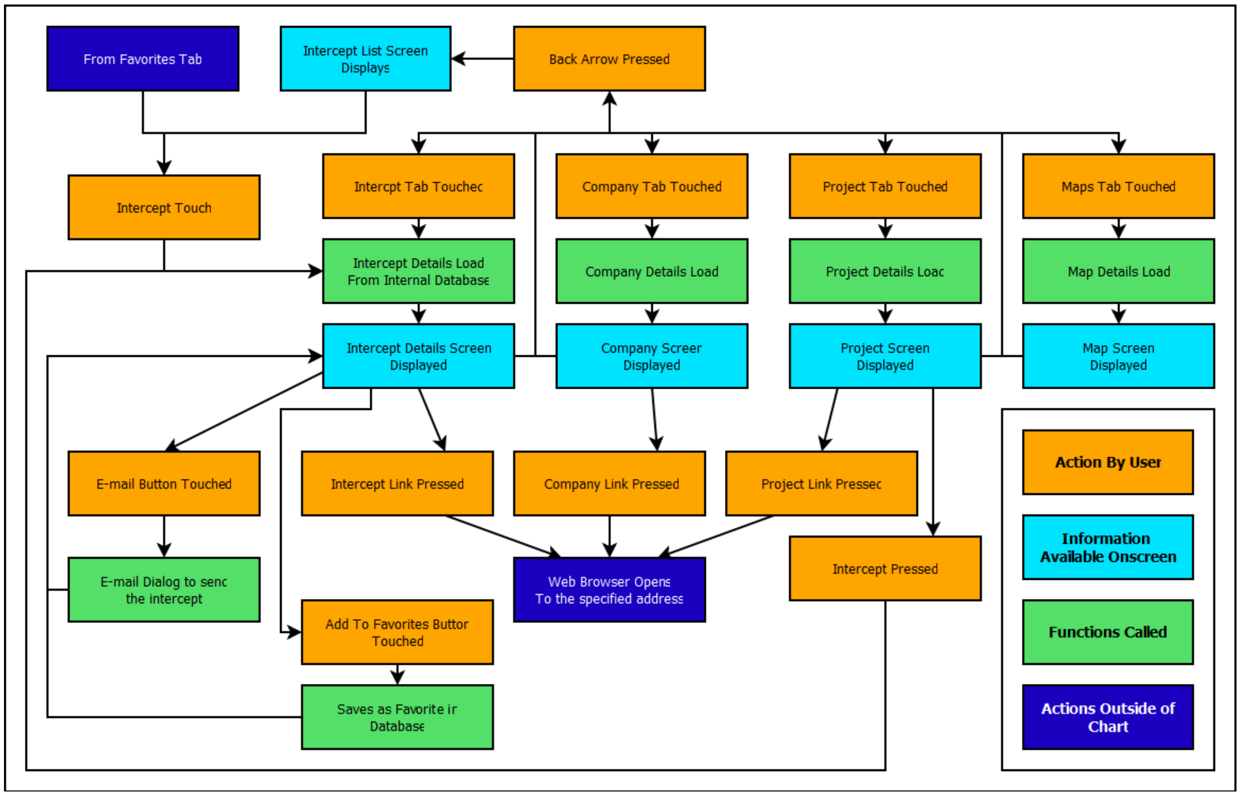


Figure 14. Screens Activity Diagram

VIII. Glossary

- Android - an operating system, usually running on mobile devices
- API (Application Programming Interface) - an application that communicates with other applications
- Intercept - geological data received from a drillhole (i.e. a core sample)
- Project - a grouping of intercepts, drilled at one site by a single company
- SQLite - an embedded relational database management system (used in the Android operating system)
- XML (Extensible Markup Language) - a set of rules for encoding documents in machine-readable form

IX. References

Android Developer Website
http://developer.android.com/index.html

EasyMock
http://easymock.org/

Hot Drillholes Database Wiki (requires Newmont login credentials)
http://174.129.192.184/wiki/index.php/Hot_Drillholes_Database