



**Matthew Harris**

**Senior Project**

**Proposal**

**getNote**

**The Mobile Application**

## Table of Contents

<i>Background and Motivation</i>	3
<i>Problems to be Addressed:</i>	3
<i>Solution:</i>	4
<i>Methodology:</i>	4
<i>Goal:</i>	4
<i>Deliverables:</i>	5
<i>Justification:</i>	5
<i>Non-Exclusive Limited IP Granting to Department:</i>	5
<i>References:</i>	6

**Student:** Matthew Harris

**Project:** getNote<sup>1</sup> Android<sup>2</sup> application

**Collaborators:** Dr. Melody Stapleton

**Client:** getNote

**Users:** Students studying computer science.

### **Background and Motivation:**

In CINS 465 my group and I created an online note taking/sharing website. After that semester I inherited the project and modified it to allow anyone with a .edu email to join the site. This allows students to take notes online for their computer science classes, and view notes taken by other students. This service only works if the user has a computer that is connected to the Internet. I would like to expand the site to have a mobile application, which would have an offline mode for their notes.

### **Problems to be addressed:**

- ⤴ Learning and creating an Android application
  - Creating a SQLite<sup>3</sup> database
  - Querying, inserting, selecting, removing from SQLite® database
  - Managing files on the mobile device
  - Connecting to the device hardware
    - Camera, Storage
  - Creating a friendly user interface (UI)
- ⤴ User authentication with pre-existing backend (MySQL<sup>4</sup>, PHP<sup>5</sup>)
  - Implementing JSON (JavaScript Object Notation) and HTTP Post / Get
- ⤴ Creating backend PHP© methods for connecting device to web server
- ⤴ Syncing device to backend including some form of text file merge

---

<sup>1</sup> getNote is a registered domain and the Intellectual Property of Matthew Benjamin Harris

<sup>2</sup> Android is a registered trademark of google Inc. [1]

<sup>3</sup> SQLite is in the public domain. [11]

<sup>4</sup> MySQL “The world’s most popular open source data base” [12]

<sup>5</sup> Copyright © 2001 – 2011 The PHP Group [13]

### **Solution:**

Once the application is opened on the mobile device, it will allow users to save their username and password for authenticating through the website database for syncing. A user will be capable of editing and creating notes. They will be able to take photos from their devices as notes, as well as view their old notes. I would like to keep a copy of all the notes on the device so that the application would work offline. Once the application is online and connected to the server it would compare all the text notes' modification dates and update whichever database was outdated, also updating if one is missing a note the other has.

To tie this application in with the website's backend I will have to learn and implement some JSON and HTTP service request. This is needed since the application will be in Java with an SQLite® database, which, once connected to the Internet, will be connecting, and passing information back and forth, with my Linux<sup>6</sup>, Apache<sup>7</sup>, MySQL®, and PHP© (LAMP) backend.

### **Methodology:**

I will be coding in Eclipse<sup>8</sup> using the Android® Software Development Kit (SDK). I will be storing the code base for the project on projectlocker.com<sup>9</sup>. I will be using the Android® emulator for development testing as well as pushing working builds to an actual device(my and friends devices).

### **Goal:**

My goal is to deploy an effective, user-friendly Android® application for note sharing.

---

<sup>6</sup> Copyright © 2011 Linux.com. [18]

<sup>7</sup> Copyright © 2011 The Apache Software Foundation [19]

<sup>8</sup> Copyright © 2011 The Eclipse Foundation [14]

<sup>9</sup> Copyright © 2011 ProjectLocker [15]

### **Deliverables:**

I will deliver all required documentation by the course advisor, and my faculty advisor. I will deliver a code complete application, with a fully functional backend that is well commented.

### **Justification:**

getnote.org is a very useful website for students, but has many limitations. In today's classrooms and study groups more and more students are taking pictures of white boards with their Android devices. This project would give them a way to organize them, furthermore a quick and easy way to upload their text notes. Also being able to easily view their notes would allow for easier review, as they would not have to take out a laptop or even need an Internet connection. In addition, teaching myself Android® will help me in the ever-changing job market when I graduate.

What I would like to take away from this project is a better understanding of the Android® SDK and web services using JSON and HTTP in regards to mobile development. I will be more proficient at the use of the IDE Eclipse®.

### **Non-Exclusive Limited IP Granting to Department:**

The authors of this work hereby grant non-exclusive and limited publication rights to the department and university to freely display the project summary and/or poster within the physical confines of the university and/or on its web presence. We also grant non-exclusive and limited academic usage of the project documents, for course demonstration purposes.

## References:

- [01] Haseman, Chris. Android Essentials. Berkeley, CA. 2009 [16]
- [02] Myrphy, Mark. Beginning Android. Berkeley, CA. 2009 [16]
- [03] <http://www.android.com/>
- [04] <http://androidforums.com/>
- [05] <http://onthefencedevelopment.com/?p=455>
- [06] <http://en.wikipedia.org/wiki/SOAP>
- [07] <http://www.helloandroid.com/tutorials/connecting-mysql-database>
- [08] [http://www.anddev.org/working\\_with\\_the\\_sqlite-database\\_-\\_cursors-t319.html](http://www.anddev.org/working_with_the_sqlite-database_-_cursors-t319.html)
- [09] <http://stackoverflow.com/questions/tagged/android>
- [10] <http://www.enode.com/x/markup/tutorial/mvc.html>
- [11] <http://www.sqlite.org/>
- [12] <http://www.mysql.com/>
- [13] <http://www.php.net/>
- [14] <http://www.eclipse.org/>
- [15] <http://projectlocker.com/>
- [16] <http://www.apress.com>
- [17] <http://www.enode.com/x/markup/tutorial/mvc.html>
- [18] <http://www.linux.com/>
- [19] <http://www.apache.org/>
- [20] Definition for wifi: Wireless local area network: a local area network that uses high frequency radio signals to transmit and receive data over distances of a few hundred feet; uses Ethernet protocol. [wordnetweb.princeton.edu/perl/webwn](http://wordnetweb.princeton.edu/perl/webwn)



**Matthew Harris**

**Senior Project**

**Project Plan**

**getNote**

**The Mobile Application**

## **Table of Contents:**

<i>Summary:</i>	9
<i>Tasks – Development:</i>	9
<i>Tasks – Paperwork:</i>	12
<i>Charts:</i>	13
<i>Dependencies &amp; Resources:</i>	15
<i>Risk Management &amp; Mitigation Strategies:</i>	16
<i>Methodology:</i>	17
<i>Non-Exclusive Limited IP Granting to Department:</i>	18
<i>References:</i>	19
<i>Web Sites Viewed:</i>	19



**Student:** Matthew Harris

**Project:** getNote<sup>10</sup> Android<sup>11</sup> application

**Collaborators:** Dr. Melody Stapleton

**Client:** getNote

**Users:** Students studying computer science.

## Summary

I will be creating an Android™ application that will allow users to take and view their class notes on their Android™ device, in addition to allowing them to sync their text notes with the pre-existing web server.

## Tasks – Development

### Updating getNote:

First I will be updating getnote.org to allow users to upload .jpg files to allow for the Android™ application to sync with the web server, when that time comes.

Due to the concern of cheating only text notes will be uploaded at this time.

Second I will also be updating the way notes are flagged for offensive or copyrighted materials. As it stands there is no way to flag a .pdf or .jpg. Lastly I will be updating getnote.orgs database to save modification date of files so when the time comes I can compare the last modified date of a file on the server and on the device, to determine which is out of date.

---

<sup>10</sup> getNote is a registered domain and the Intellectual Property of Matthew Benjamin Harris

<sup>11</sup> Android is a registered trademark of google Inc.

### Creating SQLite Database:

Using the Android™ SDK to write an SQLite<sup>12</sup> database class that will create my applications database the first time the application is run on any device.

### Querying SQLite Database:

Using the model controllers I will plan out and create queries needed for the application.

### Managing Data:

Using the Android™ SDK to create a notes directory for photo notes, text notes will be stored in the SQLite Database.

### Device Hardware:

Using the Android™ SDK to make request to the device to use the camera widget for saving new photos, I will also have to make a request to the system for displaying and selecting photos that have already been taken and saved.[6][7]

### User Interface (UI):

With the user experience in mind, I will be creating a UI that is easy and understandable, in order to create an application the user will enjoy using.

---

<sup>12</sup> SQLite is in the public domain.

### Backend:

I will be creating three of PHP<sup>13</sup> backend helper files, to allow users' devices to connect to the backend while connected to the Internet through JSON and HTTP request.

### JSON (JavaScript Object Notation)

I will have to create Java<sup>14</sup> methods with the Android™ SDK to implement some type of secure connection to the backend for sending and receiving data.

### User Authentication:

Once the backend PHP© helper functions are in place and I have the HTTP Java© methods I will test and implement user login with the backend with JSON. If the user does not have an account with the backend, they will be asked if they would like to create one and given a link to create an account.

### Syncing device:

If the device is online and the user has saved a valid account. The user will be able to press sync which will send all there device notes to the server to compare the notes on device and the backend. If there is a file with the same name on both, with different last modified dates, It will update whichever is out of date.

---

<sup>13</sup> PHP is copyrighted © 2001 – 2011 to *The PHP Group*

<sup>14</sup> Java is copyrighted © 2011 to *ORACLE Inc.*

## Tasks – Paperwork

### Project Proposal:

Dr. Melody Stapleton approved this form on August 31, 2011. A copy can be viewed here: [http://www.mattben.info/csci490/Project\\_Proposal.pdf](http://www.mattben.info/csci490/Project_Proposal.pdf)

### Project Plan:

This is the Project Plan and once approved by Dr. Melody Stapleton it will be viewable here: [http://www.mattben.info/csci490/Project\\_Plan.pdf](http://www.mattben.info/csci490/Project_Plan.pdf)

### Midterm Documentation:

I will be submitting an Entity Relationship Model (ERD) of the SQLite® database; A Unified Modeling Language (UML) diagram for the Java© classes used in thus far for the Android™ application; along with a Use Case Diagram with use case explanations.

### Poster (draft form):

Dr. Melody Stapleton will review a prototype of the poster I will be presenting and a game plan for the poster presentation will be created.

### Poster:

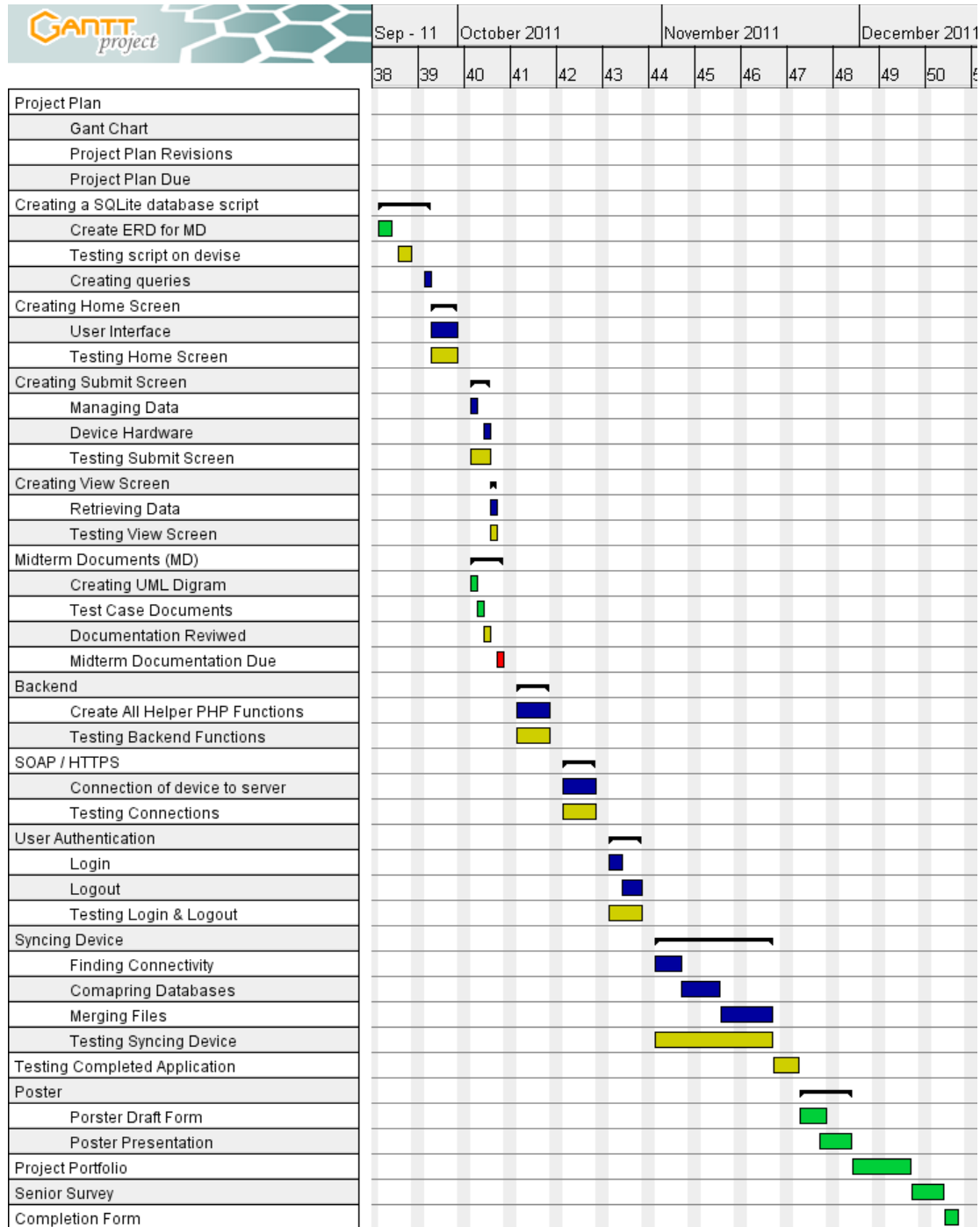
I will present and explain my project to all faculty and students to the best of my ability, as is expected of me.

### Project Completion Form:

The Project Completion Form will be submitted with my project portfolio, which will include all documentation listed before this.

## Charts

(Documentation, Development, Testing, Due Dates)





## Dependencies & Resources

### Device:

I am dependent on the use of my Motorola<sup>15</sup> Droid<sup>16</sup> 3 since it will be used to show off my work at the poster presentation along with my final presentation.

### Linode:

I am dependent on Linode<sup>17</sup> since my web server and backend are hosted there, as this is a service I pay for.

### Daniel Phelps:

I am using Daniel Phelps' documentation as a resource to help structure my documentation.

### Ben Carlsson

I am having Ben Carlsson proofread my documentation. He is a key resource to help ensure high quality in my project documentation.

### getNote.org (noteZilla):

Thank you to Mike Wood, Jimmy Weir, and Kevin Kane, as the backend website <http://www.getnote.org> was started as a group project in our Web Development (CINS 465) class. The full documentation for our project can be viewed here: <http://www.mattben.info/csci465/Milestone7.pdf>. I adopted this project, then known as noteZilla, and made it my own, modifying the use of the site as well as moving and renaming it to getNote.org.

---

<sup>15</sup> Motorola is copyrighted ©2011 to *Motorola Mobility Inc.*

<sup>16</sup> DROID is a registered trademark of *Lucasfilm Ltd*

<sup>17</sup> Linode is Copyrighted © 2003 – 2011 to *Linode LLC*

## Risk Management & Mitigation Strategies

<p><u>Learning Curve:</u></p> <ul style="list-style-type: none"> <li>I am the only student teaching himself or herself the Android™ SDK this semester. There is not a faculty member with experience creating mobile applications using the Android™ SDK. I have a concern that I will not learn the Android™ SDK as fast I believe I will, and that I will not have all the features completed in time.</li> </ul>	<ul style="list-style-type: none"> <li>✦ I will meet with Chris Morris, since he has experience with Apple<sup>18</sup> iOS<sup>19</sup> mobile development.</li> <li>✦ I will be starting development as soon as possible and have already started reading books on creating Android™ applications.</li> </ul>
<p><u>Resources:</u></p> <p>I am dependent on the use of my:</p> <ul style="list-style-type: none"> <li>Computer</li> <li>Project Locker<sup>20</sup> account</li> <li>Motorola© Droid™ 3 device</li> <li>Linode© server cluster</li> </ul>	<ul style="list-style-type: none"> <li>I will have copies of my project and project files in many places.</li> <li>I have a back up computer, which can be used as a fallback.</li> <li>My wife and many friends have Android phones I could borrow in a bind.</li> <li>I have daily backups of getNote.org. I could push the site to my local machine or the school server.</li> </ul>
<p><u>Writing Ability:</u></p> <ul style="list-style-type: none"> <li>I am a weak writer and will have difficulty getting all the writing requirements done on time.</li> </ul>	<ul style="list-style-type: none"> <li>I have to start all the writing requirement documents early so that I can have them reviewed.</li> <li>Thank you to Ben Carlsson for all the proofreading and comments.</li> </ul>
<p><u>Self Control:</u></p> <ul style="list-style-type: none"> <li>I get excited about my projects, and can tend to feature creep or get off task.</li> </ul>	<ul style="list-style-type: none"> <li>I am going to keep in close contact with my advisor, Dr. Melody Stapleton, and adhere to this project plan.</li> </ul>

<sup>18</sup> Apple is Copyrighted © 2011 to *Apple Inc.*

<sup>19</sup> iOS (iPhone OS) is a registered trademark of *Apple Inc.*

<sup>20</sup> Copyright © 2011 *ProjectLocker*



## Methodology

### Programming Paradigm:

Java© follows the object-oriented paradigm, and the same holds true when using the Android™ SDK.

### Languages & Tools:

I will be writing getNote the Android application in Java©, the primary language that implements the Android™ SDK. This development will be taking place in Eclipse<sup>21</sup>, an Integrated Development Environment (IDE). I have chosen Eclipse© over other the other IDEs for the power and ease of Eclipse© and its auto completion. Eclipse© also has the most support for the Android™ SDK, which makes for easier testing and debugging. [6][7] I will be writing the backend helper files in PHP© and, though there are many powerful and useful IDEs for creating PHP©, I am most comfortable using a simple text editor Notepad++<sup>22</sup>. I will be pushing these files to my web server using WinSCP<sup>23</sup>. I do all my web development testing in the web browser Mozilla<sup>24</sup> Firefox<sup>25</sup>. I will be making all changes to the backend MySQL® database using phpMyAdmin<sup>26</sup>. I will be keeping copies of all my files and project source code in my Project Locker© folder as well as locally and on my backend server.

---

<sup>21</sup> Eclipse is Copyrighted ©2011 to *The Eclipse Foundation*

<sup>22</sup> Notepad++ is Copyrighted © 2011 to *Don Ho*

<sup>23</sup> WinSCP is free software which falls under the GNU General Public License

<sup>24</sup> Mozilla is copyrighted ©1998 – 2011 to *mozilla.org*

<sup>25</sup> Firefox is a registered trademark of *mozilla.org*

<sup>26</sup> phpMyAdmin is copyrighted ©2003 – 2011 to *phpMyAdmin devel team*

### Development Approach:

I have read over a few approaches. I will be using a mix of things. The first and foremost approach will be Rapid Application Development (RAD) [1]. I feel this is a good approach for me since I'm new to Android™ and completing a small part and then testing and reviewing it, and then repeating this process I believe this will be a good way for me to keep the project under control. I will be using some of the features of Extreme Programming (EX) [3]; again, the rapidness of pushing and reviewing builds is a key. These all tie in to the Holy Grail of development paradigms: Agile Programming [2], the idea of breaking the project up into milestones (or builds) and then breaking these milestones into iterations (or features) [2]. I feel this kind of approach will lead me to a successful Android™ application come December.

### Test Plan:

Since I will be using a rapid development approach [1] all my friends and I will be manually testing each feature as it is added. This will result in each build containing a new feature and bug / logic fixes from previous builds.

### Non-Exclusive Limited IP Granting to Department:

The authors of this work hereby grant non-exclusive and limited publication rights to the department and university to freely display the project summary and/or poster within the physical confines of the university and/or on its web presence. We also grant non-exclusive and limited academic usage of the project documents, for course demonstration purposes.

## References:

- Fig 1.1      <http://www.enode.com/x/markup/tutorial/mvc.html>  
<http://stackoverflow.com/questions/2925054/mvc-pattern-in-android>
- [1]            CASEMaker Totem; What is Rapid Application Development?  
Copyright © 1997-2000 by CASEMaker Inc.  
[http://www.casemaker.com/download/products/totem/rad\\_wp.pdf](http://www.casemaker.com/download/products/totem/rad_wp.pdf)
- [2]            Pilone, Dan & Miles, Russ; Head First Software Development  
Copyright © 2008 by O'Reilly Media, Inc.
- [3]            Beck, Kent; eXtreme Programming eXplained  
Copyright © 2000 by Addison Wesley Longman, Inc.
- [4]            Kulak, Daryl & Guiney, Eamonn; Use Cases Requirements in Context 2<sup>nd</sup> Edition  
Copyright © 2004 by Pearson Education, Inc.
- [5]            Stiller, Evelyn & LeBlanc, Cathie; Project-Based Software Engineering  
Copyright © 2002 by Addison-Wesley
- [6]            Haseman, Chris; Android Essentials <http://www.apress.com>  
Copyright © 2008 by Chris Haseman
- [7]            Murphy, L Mark; Beginning Android <http://www.apress.com>  
Copyright © 2009 Mark L Murphy

## Web Sites Viewed:

- {1}            <http://www.android.com/>
- {2}            <http://en.wikipedia.org/wiki/SOAP>
- {3}            <http://stackoverflow.com/questions/tagged/android>
- {4}            <http://www.enode.com/x/markup/tutorial/mvc.html>
- {5}            <http://www.sqlite.org/>
- {6}            <http://www.mysql.com/>
- {7}            <http://www.php.net/>
- {8}            <http://www.eclipse.org/>
- {9}            <http://www.dropbox.com/>
- {10}           <http://www.apress.com>
- {11}           <http://www.linux.com>



**Matthew Harris**

**Senior Project**

**Midterm Documentation**

**getNote**

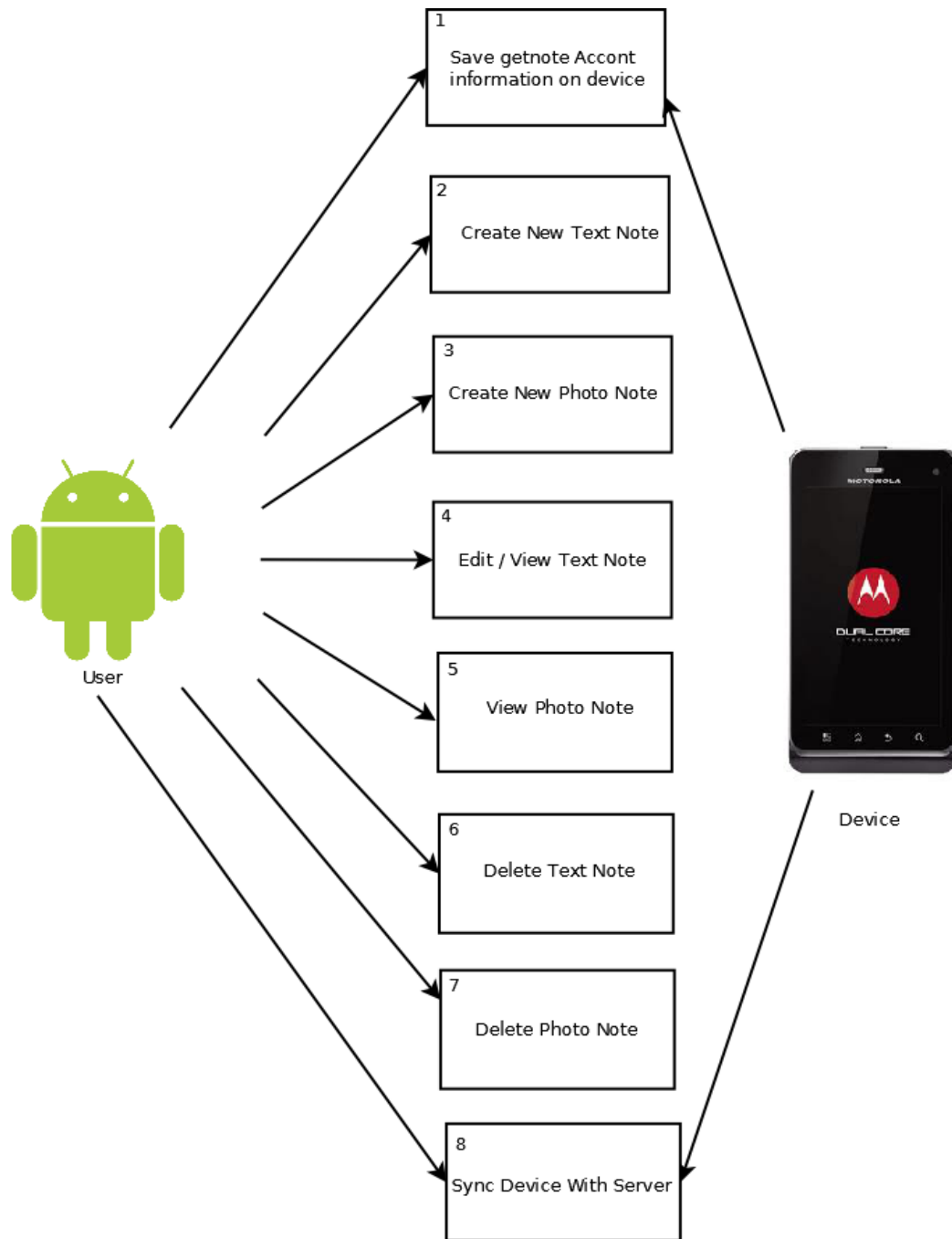
**The Mobile Application**

## Table of Contents

<i>Use Case Diagram</i>	22
<i>Use Case Seniors / User Manual</i>	23
<i>UML Class Diagram Introduction</i>	26
<i>File System</i>	26
<i>UML Class Diagram</i>	27
<i>Android® Class Diagram</i>	28
<i>References</i>	29
<i>Midterm Presentation Slides</i>	30

**Student:** Matthew Harris  
**Project:** getNote<sup>27</sup> Android<sup>28</sup> application  
**Collaborators:** Dr. Melody Stapleton  
**Client:** getNote  
**Users:** Students studying computer science.

### Use Case Diagram



<sup>27</sup> getNote is a registered domain and the Intellectual Property of Matthew Benjamin Harris

<sup>28</sup> Android is a registered trademark of google Inc.

## Use Case Scenarios / User Manual

### 1. Save getNote account information on the device

- Actor: User
  - ✕ The first time the user opens the getNote application, they will be taken to a screen asking them to enter their school email and password (the same as on the server). The user will then press submit, and they will be taken to the *home screen*.
- Actor: Device
  - ✕ Once the user has entered in their username and password, and the device has connection, the device will call the *lookup service* which will pull all the users notes off the server and save them on the device.

### 2. Create New Text Note

- Actor: User
  - ✕ The user starts on the *home screen*, and will select the “menu/settings” button on their device, this will bring up the application menu, They will select “new note” and it will open the *class information screen*. The user will then enter the class, class type, and lecture date for their note. They will then press the Text Note button which will open the *text editor screen* where the user will type in their note. Once they are done they will press the submit button. At this point the note will be saved on the device.

### 3. Create New Photo Note

- Actor: User
  - ✕ The user starts on the *home screen*, and will select the “menu/settings” button on their device, this will bring up the application menu, They will select “new note” and it will open the *class information screen*. The user will then enter the class, class type, and lecture date for their note. They will then press the Photo Note button, which will call the devices *camera widget* (the pre-installed camera handler). It will ask if they want to select a photo that is on the device or take a new one. Once they have selected or taken a photo, the camera widget will hand back a pointer to the photo and the device will save the photo.

### 4. Edit / View Text Note

- Actor: User
  - ✕ The user starts on the *home screen*, and will select (press) one of the pre-existing text notes listed. This will open the *text editor screen* where the user will review or edit the note. Once they are done they will press the submit button if changes were made. Or simply press the back button on the device to return to the *home screen*.

### 5. View Photo Note

- Actor: User
  - ✕ The user starts on the *home screen*, and will select (press) one of the pre-existing photo notes listed. This will launch the devices gallery widget which will allow the user to see the photo, once they are done they will select the device back button which will return them to the *home screen*.



## 6. Delete Text Note

- Actor: User
  - ✕ The user starts on the *home screen*, and will select (press) one of the pre-existing .txt notes listed, they will then be taken to the *text editor screen*. The user will then press the “Delete Note” button, this will delete the note. The user will be returned to the *home screen* and the note list will be update.

## 7. Delete Photo Note

- Actor: User
  - ✕ The user starts on the *home screen*, and will select (press) one of the pre-existing .jpg notes listed, they will then be taken to the device's gallery widget . The user will select the “menu/settings” button on their device this will bring up the application menu, the user will then select “Delete”, this will delete the note. The user will be returned to the *home screen* and the note list will be update.

## 8. Sync Device With Server

- Actor: User
  - ✕ The user starts on the *home screen*, and will selects the “menu/settings” button on their device this will bring up the application menu, the user will then select “sync”. This will update the list of notes on the *home screen* and/or the server.
- Actor: Device
  - ✕ The device will call the lookup service every time there is connectivity to sync the device with the server. Pulling and pushing files to and from the server.

## UML Class Diagram Introduction

Thus far I foresee creating Six java classes, four activities, one service, and one helper class for the SQLite<sup>29</sup> database. In the Android® SDK classes that extend activity are the screens the user interacts with, while class that extend service run in the background, which are not seen by the user but can be used by a user, an example would be when a user wants to update their list of notes from the server. The screen does not change only updates. My UML Class Diagram is on the next page (page 8).

## File System

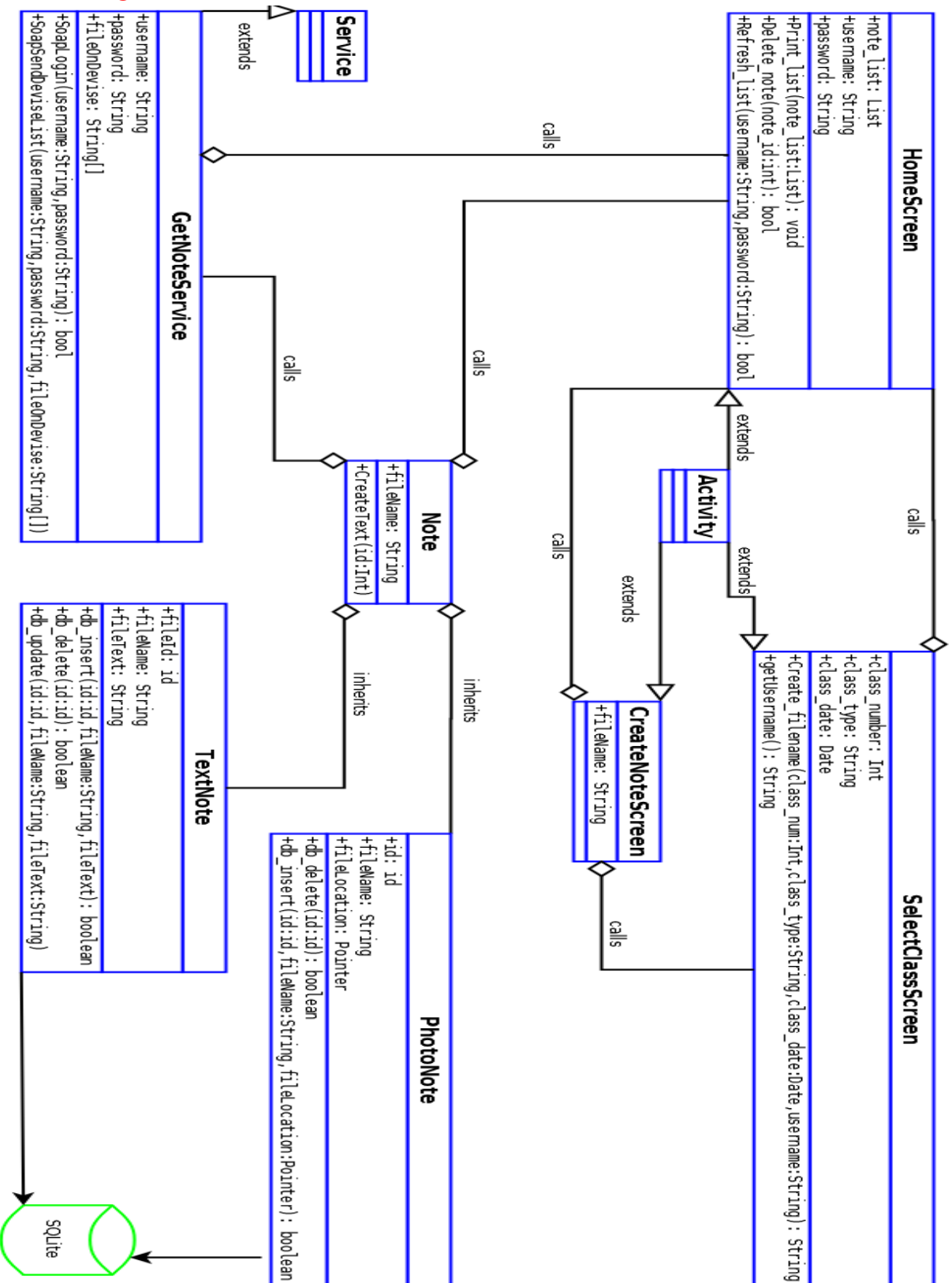
Through out my reading of the Android website[1] and my Android books[2][3] I have learned that SQLite is not a replacement or alternative for a MySQL<sup>30</sup> database[2]. Its a quicker way of fopen a indexed file system[3]. The most common set up for a SQLite® database is one table with three attributes id, file name, and file content[3]. This works well with saving the notes but the little things like the list of class' and course types which are stored in the sites database are stored in the application string file which can be updated dynamically. One down side of this is that I don't have a ER diagram rather a table of index's for my files.

Notes	
* <u>id</u>	<u>id</u>
•fileTitle	Varchar(128)
•fileText	Varchar(2048)

---

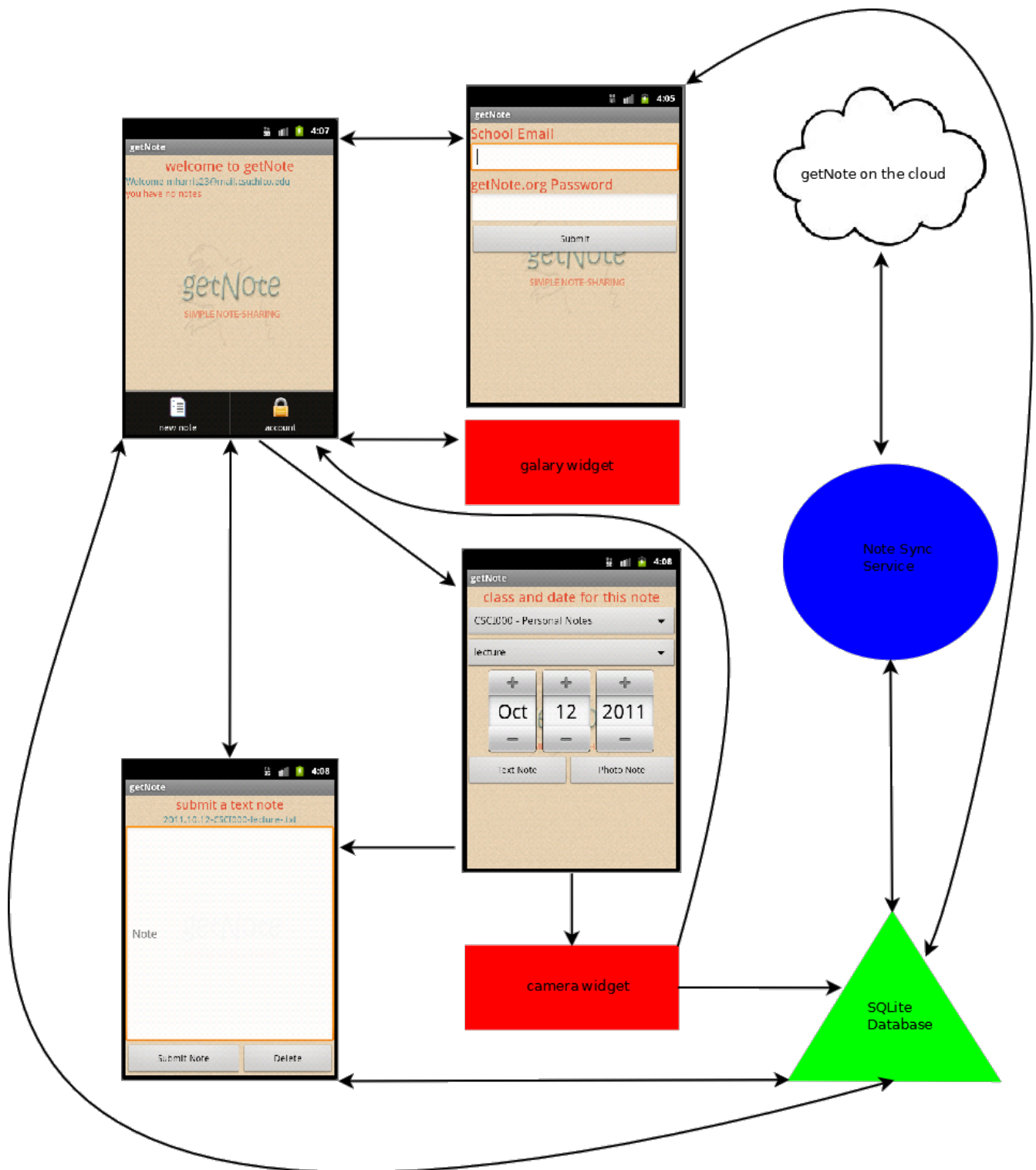
<sup>29</sup> SQLite is in the public domain.

<sup>30</sup> MySQL is in the public domain.



UML Class Diagram

## Android® Class Diagram



## References:

- [1] <http://developer.android.com/index.html>  
<http://developer.android.com/reference/android/database/sqlite/package-summary.html>  
Copyright © 2011 google Inc.
- [2] Haseman, Chris; Android Essentials <http://www.apress.com>  
Copyright © 2008 by Chris Haseman
- [3] Murphy, L Mark; Beginning Android <http://www.apress.com>  
Copyright © 2009 Mark L Murphy

## getNote


**Student:** Matthew Harris  
**Advisor:** Dr. Melody Stapleton  
**Client:** <http://www.getnote.org/>  
**Acknowledgment:** noteZilla Team  
 Kevin Kane, James Weir, Michael Wood  
**Project:** Android® application




CS231 S23, Spring, Professor, CS23C CALIFORNIA STATE UNIVERSITY, CHICO

## Project Description



**Goal:** My goal is to deploy an effective, user-friendly Android® application for note sharing.




**Users:** CSU Chico CSCI Students  
**Features:** See Use Case Diagram

CS231 S23, Spring, Professor, CS23C CALIFORNIA STATE UNIVERSITY, CHICO

## Preliminary Investigations



CS231 S23, Spring, Professor, CS23C CALIFORNIA STATE UNIVERSITY, CHICO

## Anticipated METHODOLOGIES & TOOLS

**Framework**

- Android® SDK

**General Deployment Plan**

- Rapid Development

**General Test Plan**

- Self & Users

**Development Platform**

- Eclipse

**Development Phases**

- Iterations (4)



CS231 S23, Spring, Professor, CS23C CALIFORNIA STATE UNIVERSITY, CHICO

## Challenges & Choices

**Significant challenges or obstacles**


- Teaching myself the Android® SDK
- Time Management
- Feature Creep

**Choices I made and reasoning**

- File name holds all note information, due to some limitations of the web server.

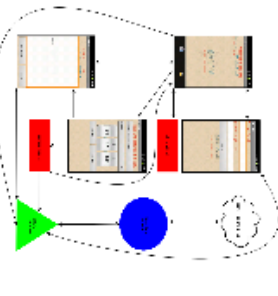
**Limitations posed by my approach**

- Users will only come from CSU Chico



CS231 S23, Spring, Professor, CS23C CALIFORNIA STATE UNIVERSITY, CHICO

## STATUS



CS231 S23, Spring, Professor, CS23C CALIFORNIA STATE UNIVERSITY, CHICO



**Matthew Harris**

**Senior Project**

**Poster | Presentation Slides | Source Code**

**getNote**

**The Mobile Application**

## Table of Contents

<i>Project Poster</i>	33
<i>Presentation Slides</i>	34
<i>sync.php</i>	35
<i>json.php</i>	39
<i>isuser.php</i>	40
<i>CreateNoteActivity.java</i>	41
<i>HomeActivity.java</i>	45
<i>Login.java</i>	57
<i>NotesDbAdapter.java</i>	63
<i>PoliciesActivity.java</i>	67
<i>SelectionActivity.java</i>	68
<i>AndroidManifest.xml</i>	77
<i>home.xml</i>	78
<i>input.xml</i>	79
<i>login.xml</i>	80
<i>notes_row.xml</i>	81
<i>policies.xml</i>	81
<i>submit.xml</i>	82
<i>text1.xml</i>	83
<i>menu.xml</i>	83
<i>strings.xml</i>	84



**Student:** Matthew Harris

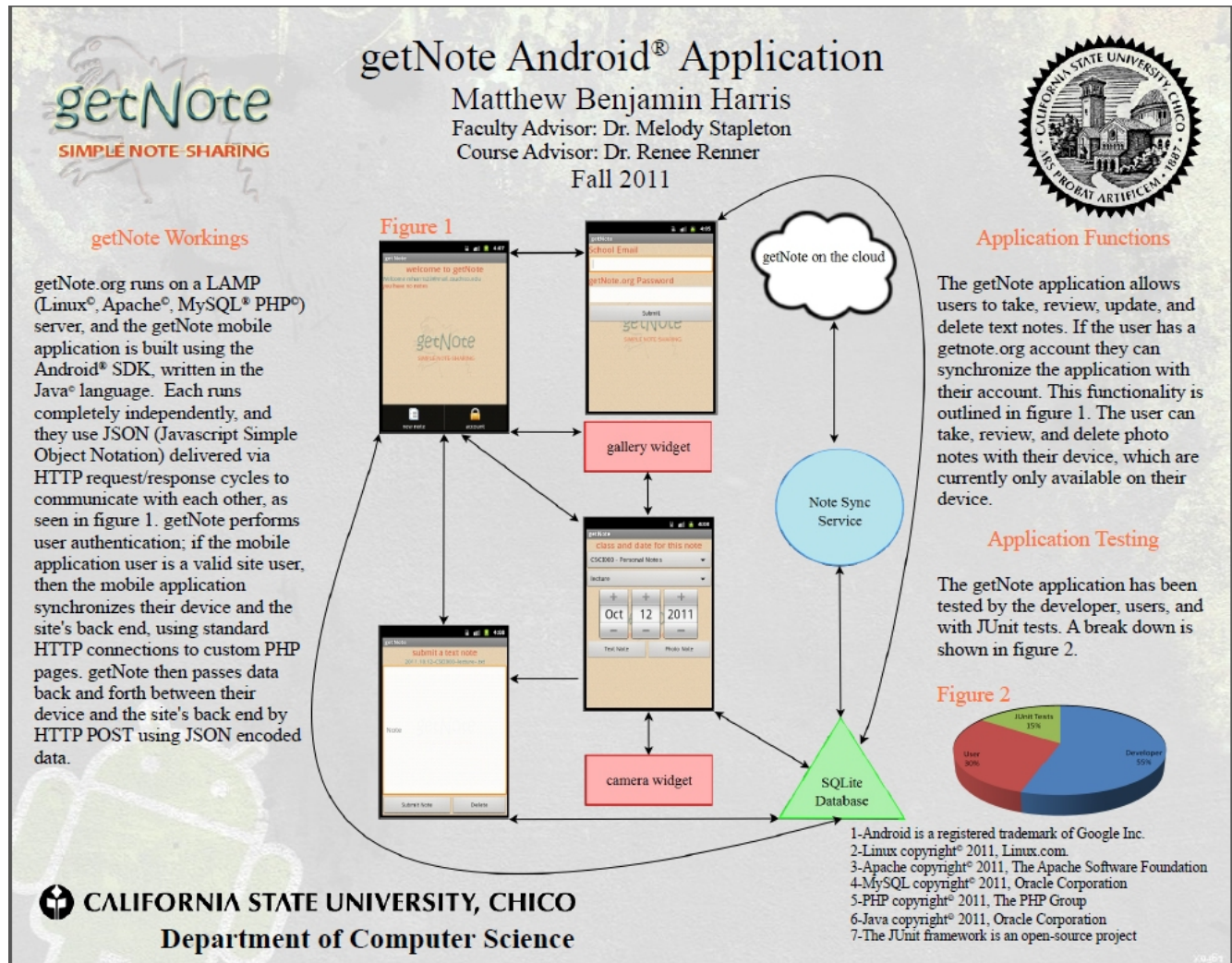
**Project:** getNote<sup>31</sup> Android<sup>32</sup> application

**Collaborators:** Dr. Melody Stapleton

**Client:** getNote

**Users:** Students studying computer science.

**Project Poster:**




<sup>31</sup> getNote is a registered domain and the Intellectual Property of Matthew Benjamin Harris

<sup>32</sup> Android is a registered trademark of google Inc. [1]

1

# getNote Android® Application

Matthew Benjamin Harris  
Faculty Advisor: Dr. Malady Sridelton  
Course Advisor: Dr. Kenes Kemner  
Fall 2011



getNote  
SHIRAZ MOHAMEDHARUN

http://www.getnoting.org

Slide 1

2

## Motivation | Background

**Special Thanks:**  
To the computer scientists using the app, to the students who helped with the app, to the faculty and staff who supported the app, to the friends who helped with the app, to the family who supported the app, to the world who supported the app.

http://www.getnoting.org

Slide 2

3

## Related Things Research | Products

**Research**  
Android®  
Java®  
JSON  
HTTP  
Web Service

**Similar Products**  
Microsoft OneNote®  
EverNote®  
Google Docs



http://www.getnoting.org

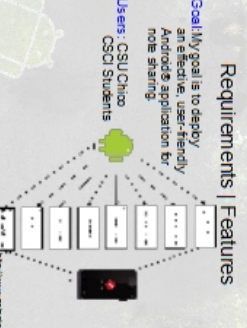
Slide 3

4

## Requirements | Features

Goal: My goal is to develop an Android user-friendly note sharing application for note sharing.

Users: CSU Chico  
CSU Students



http://www.getnoting.org

Slide 4

5

## Design Issues | Decisions | Choices

**Significant challenges or obstacles:**

- Teaching myself the Android SDK
- Learning how web requests work
- Interaction between device, server through HTTP and passing JSON strings

**Choices: I made and reasoning:**

- File name holds all note information.

**Limitations posed by my approach:**

- Users will only come from CSU Chico



http://www.getnoting.org

Slide 5

6

## Methods

**Framework:**

- Android® SDK

**General Deployment Plan**

- JSON
- HTTP request

**General Test Plan**


- Self & Users

**Development Platform**

- PHP®
- MySQL®
- SQLite®

**Development Phases**

- Iterations (4)





http://www.getnoting.org

Slide 6

7

## Testing | Test Plan

- I downloaded and researched automated testing with the JUnit framework from <http://www.junit.org>
- The app is available for download from the homepage. I wrote a form for user feedback which was very helpful.

http://www.getnoting.org

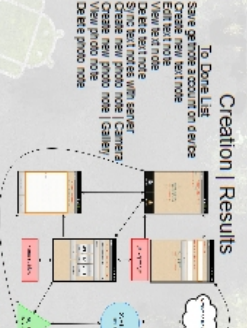
Slide 7

8

## Creation | Results

**To Do List:**

- Save get note account on device
- Create new note
- Edit note
- Delete note
- Sync note with server
- Create new photo note (Camera)
- View photo note (Gallery)
- Delete photo note



http://www.getnoting.org

Slide 8

9

## Analysis | Summary | Future Work

**To Do List:**

- Test notes for quick look ups
- Have online live update from server
- Add more classes imports
- Add major related imports to test notes

**Phone® app needed next**

More users needed



http://www.getnoting.org

Slide 9

10

## DEMO



http://www.getnoting.org

Slide 10

11

## Questions?



http://www.getnoting.org

Slide 11

12

## Citations

Android is a registered trademark of Google Inc.  
Java copyright 2011 Oracle Corporation  
Microsoft One is copyrighted 2011 Microsoft Inc  
EverNote is copyrighted 2011 Evernote Corporation  
Google Docs is copyrighted 2011 Google Inc  
iPhone is a registered trademark of Apple Inc

http://www.getnoting.org

Slide 12

&lt;?php

```

include_once('../db.php');

$check = $_POST['json'];
$json = json_decode($check);
$sync = $json->{'sync'};
$blank = false;
if (empty($sync))
{
    $blank = true;
}

if($blank)
{
    echo "blank";
}
else
{
    $notes = explode("[", $sync);
    $size = count($notes);

    $username = $notes[0];
    $username = explode("-", $username);
    $username = $username[1];

    $fileName = array();
    $fileBody = array();

    db_open();
    $result = mysql_query("SELECT * FROM note WHERE author='$username'");
    while($row = mysql_fetch_array($result))
    {
        $serverFiles[] = $row[0];
        $serverDates[] = $row[3];
    }

    for($p = 0; $p < $size; $p++)
    {
        if($p%2 == 0)
        {
            $file1=explode("-", $notes[$p]);
            $modDate1 = $file1[0];
            $user1 = $file1[1];
            $lec_date1 = $file1[2];
            $class1 = $file1[3];

```

```

        $type1 = $file1[4];
        $ext1 = $file1[5];
        //user-date-class-type-ext
        $title1 = $user1 . "-" . $lec_date1 . "-" . $class1 . "-" . $type1 . "-" . $ext1;
        $Phonenotes[] = $title1;
    }
}

$CheckAdd = array_diff($ServerFiles, $Phonenotes);
$counter = count($CheckAdd);

for($g = 0; $g < $counter; $g++)
{
    //echo "here";
    $title = $CheckAdd[$g];
    $passname = str_replace("-", ".", $ServerDates[$g]) . "-" . $title;
    $FileName[] = $passname;
    $file = "../saved_notes/" . $title;
    $FileBody[] = file_get_contents($file);
}

$file;
$modDate;
$title;
$user;
$save = false;
$send = false;

for($i = 0; $i < $size; $i++)
{
    $body;
    if($i%2 == 0) //even
    {
        $file=explode("-", $notes[$i]);
        $modDate = $file[0];
        $user = $file[1];
        $lec_date = $file[2];
        $class = $file[3];
        $type = $file[4];
        $ext = $file[5];
        //user-date-class-type-ext
        $title = $user . "-" . $lec_date . "-" . $class . "-" . $type . "-" . $ext;

        $in_table = mysql_query("SELECT filename FROM note
        WHERE filename = '$title'");

        $rw = mysql_fetch_array($in_table);
    }
}

```

```

if(empty($rw[0])) //if not in a table then insert in to the table
{
    mysql_query("INSERT INTO note(filename, lecture_date,
    creation_date, modified_date, author, course, type, flag)
    VALUES('$title', '$lec_date', '$lec_date', '$modDate', '$user',
    '$class', '$type', NULL)") or die (mysql_error());

    $body = $notes[$i+1];

    $myFile = "../saved_notes/" . $title;// now saving the note
    $fh = fopen($myFile, 'w') or die("can't open file");
    $stringData = "$body\n";
    fwrite($fh, $stringData);
    fclose($fh);
    $send = false;
}
else
{
    $in_table = mysql_query("SELECT modified_date FROM note
    WHERE filename = '$title'");

    $rw = mysql_fetch_array($in_table);
    $modDate = str_replace(".", "-", $modDate);
    $newNoteDate = explode("-", $modDate);
    $oldNoteDate = explode("-", $rw[0]);

    if($newNoteDate[0] > $oldNoteDate[0]) //year
    {
        $save = true;
        $send = false;
    }
    else if($newNoteDate[1] > $oldNoteDate[1] &&
        $newNoteDate[0] == $oldNoteDate[0]) // month
    {
        $save = true;
        $send = false;
    }
    else if($newNoteDate[2] > $oldNoteDate[2] &&
        $newNoteDate[1] == $oldNoteDate[1] &&
        $newNoteDate[0] == $oldNoteDate[0]) // day
    {
        $save = true;
        $send = false;
    }
    else if($newNoteDate[2] == $oldNoteDate[2] &&

```

```

        $newNoteDate[1] == $oldNoteDate[1] &&
        $newNoteDate[0] == $oldNoteDate[0])
    {
        $save = false;
        $send = false;
    }
    else
    {
        $send = true;
        $save = false;
    }

    if($save == true)
    {
        mysql_query("UPDATE note SET modified_date =
        '$modDate' WHERE filename = '$title'");

        $body = $notes[$i+1];

        $myFile = "../saved_notes/" . $title;// now saving the note
        $fh = fopen($myFile, 'w') or die("can't open file");
        $stringData = "$body\n";
        fwrite($fh, $stringData);
        fclose($fh);
    }
    if($send == true)
    {
        $passname = str_replace("-", ".", $rw[0]) . "-" . $title;
        $FileName[] = $passname;
        $file = "../saved_notes/" . $title;
        $FileBody[] = file_get_contents($file);
    }
}

$FileJson = array_combine($FileName, $FileBody);

print json_encode($FileJson, JSON_FORCE_OBJECT);

$myFile = "testFile.txt";
$fh = fopen($myFile, 'w') or die("can't open file");
$stringData = json_encode($FileJson, JSON_FORCE_OBJECT);
fwrite($fh, $stringData);
fclose($fh);
}
db_close();

```

?>



&lt;?php

```

include_once('../db.php');
$username = $_GET['user'];
$title = array();
$body = array();
$date = array();
$header = array('title', 'body', 'date');

db_open();
$result = mysql_query("SELECT * FROM note WHERE author='$username'");

while($row = mysql_fetch_array($result))
{
    $title[] = $row[0];
    $date[] = str_replace("-", ".", $row[3]);
}

$size = count($title);

for($j = 0; $j < $size; $j++){
    $type = explode("-", $title[$j]);
    if($type[4] != ".txt")
    {
        unset($title[$j]);
        unset($date[$j]);
    }
}
$size = count($title);

for($i = 0; $i < $size; $i++){
    $file = "../saved_notes/" . $title[$i];
    $body[] = file_get_contents($file);
}

db_close();

$setup = count($title);

for($i = 0; $i < $setup; $i++)
{
    $that = $date[$i] . "-" . $title[$i];
    $modTitle[] = $that;
}
$output1 = array_combine($modTitle, $body);
print json_encode($output1, JSON_FORCE_OBJECT);

```

?&gt;

```
<?php
include_once('../db.php');

$username = $_GET['user'];
$password = $_GET['pass'];

db_open();
// Encrypt password using MD5
$pass = md5($password);

// Escape the username to protect from SQL injection
$user = mysql_real_escape_string($username);

// Retrieve the password from the database for the specified user:
$result = mysql_query("SELECT password FROM user WHERE username='$user'");
$valid = mysql_fetch_array($result);

// Compare the given password to the stored password:
if($pass == $valid[0])
    $user = true;
else
    $user = false;

db_close();

$output = array('user' => $user);

print json_encode($output, JSON_FORCE_OBJECT);
```

```
?>
```



## CreateNoteActivity.java

```
package org.getnote;

import java.sql.Date;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.GregorianCalendar;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.text.Html;
import android.text.format.DateFormat;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class CreateNoteActivity extends Activity implements OnClickListener{

    // These are the global variables
    TextView fileNameHeader;
    EditText editSubmitNote;
    Button buttonSubmitNote;
    Button buttonDeleteNote;
    String filename;
    String body;
    boolean repeat;
    private Long mRowId;

    /*
     * (non-Javadoc)
     * @see android.app.Activity#onCreate(android.os.Bundle)
     * Called when the activity is first created.
     * This is the method that is called on start up to create what is
     * seen on the device.
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // Setting the page lay from /res/layout/submit.xml
        setContentView(R.layout.submit);
    }
}
```

```

editSubmitNote = (EditText)findViewById(R.id.editSubmitNote);
buttonSubmitNote = (Button)findViewById(R.id.buttonSubmitNote);
buttonDeleteNote = (Button)findViewById(R.id.buttonDeleteNote);
fileNameHeader = (TextView)findViewById(R.id.textViewFilename);

buttonSubmitNote.setOnClickListener(this);
buttonDeleteNote.setOnClickListener(this);

// Bundle is the things that where passed from the previous activity to this one
Bundle b = getIntent().getExtras();
filename = b.getString("filename");

fileNameHeader.setText(filename);

mRowId = null;
Bundle extras = getIntent().getExtras();

if (extras.getLong(NotesDbAdapter.KEY_ROWID) != 0) {
    // If there is a filename and filebody in the extras then set the page with that information
    Long mId = extras.getLong(NotesDbAdapter.KEY_ROWID);
    Cursor update = NotesDbAdapter.fetchNote(mId);
    mRowId = update.getLong(0);
    filename = update.getString(1);
    body = update.getString(2);

    if (filename != null){
        fileNameHeader.setText(filename);
    }
    if (body != null) {
        editSubmitNote.setText(Html.fromHtml(body));
    }
    repeat = true;
}
else{
    repeat = false;
}
}

/*
 * (non-Javadoc)
 * @see android.view.View.OnClickListener#onClick(android.view.View)
 * Depending what button is pressed this will determine what method will be called
 */
@Override
public void onClick(View arg0) {
    switch (arg0.getId()){
        case R.id.buttonSubmitNote:

```

```

// User is trying to save a note
String submitNote = editSubmitNote.getText().toString();

Calendar c=new GregorianCalendar();
Date dateD = new Date(c.getTimeInMillis());
String date = new SimpleDateFormat("yyyy.MM.dd").format(dateD);

if(!submitNote.equals("")){
    if (!repeat){
        // This was a new note, so note was saved
        long id = NotesDbAdapter.createNote(filename, submitNote,
            date);

        Toast.makeText(CreateNoteActivity.this, R.string.ToastSaved,
            Toast.LENGTH_SHORT).show();

    }
    else{
        // A note that has been updated
        boolean updated = NotesDbAdapter.updateNote(mRowId,
            filename, submitNote, date);

        Toast.makeText(CreateNoteActivity.this, R.string.ToastUpdated,
            Toast.LENGTH_SHORT).show();
    }
    // This method call tells the system we are done with this activity, and to
    // return to the previous one
    finish();
}
else{
    // User can not save a empty note
    Toast.makeText(CreateNoteActivity.this, R.string.ToastEmptyNote,
        Toast.LENGTH_SHORT).show();
}
break;
case R.id.buttonDeleteNote:
    // User is trying to delete a note, show the dialog pop up to make sure they really
    // want to delete
    final AlertDialog alertDialog = new AlertDialog.Builder(CreateNoteActivity.this).create();
    alertDialog.setTitle(R.string.AlerDialogPhotoNote);

    // Why I didn't save this next string in the /res/values/strings.xml and then call
    // alertDialog.setMessage("R.string.DialogDeleted"); this
    // Can't be used with newest API setMessage must take a char array not a pointer (int)
    alertDialog.setMessage("are you sure you want to delete");
    alertDialog.setButton("ok", new DialogInterface.OnClickListener() {
        // They click ok, then delete and let them know

```

```

public void onClick(DialogInterface dialog, int which) {
    Bundle extras = getIntent().getExtras();
    if (extras.getLong(NotesDbAdapter.KEY_ROWID) != 0) {
        Long mId = extras.getLong(NotesDbAdapter.KEY_ROWID);
        boolean del = NotesDbAdapter.deleteNote(mId);
        Toast.makeText(CreateNoteActivity.this, R.string.ToastDeleted,
            Toast.LENGTH_SHORT).show();
    }
    else{
        Toast.makeText(CreateNoteActivity.this, R.string.ToastNot,
            Toast.LENGTH_SHORT).show();
    }
    // This method call tells the system we are done with this activity, and to return to the
    previous one
    finish();
} });
alertDialog.setButton2("cancel", new DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int which) {
        alertDialog.cancel();

    } });

alertDialog.show();
break;
    }
}
}

```

## HomeActivity.java

```
package org.getnote;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.URI;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.entity.ByteArrayEntity;
import org.apache.http.entity.StringEntity;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.params.BasicHttpParams;
import org.apache.http.params.HttpConnectionParams;
import org.apache.http.params.HttpParams;
import org.json.JSONArray;
import org.json.JSONObject;
import android.app.AlertDialog;
import android.app.ListActivity;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.net.Uri;
import android.net.http.AndroidHttpClient;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;
import android.widget.TextView;
import android.widget.Toast;
```

```

public class HomeActivity extends ListActivity implements OnClickListener {

    // These are the global variables, PREFS_ are used for saving and getting, users preference from
    // the system.
    // I save the users, getNote, user name, password, account status in the MyPrefsFile which is a
    // system file.
    public static final String PREFS_NAME = "MyPrefsFile";
    private static final String PREF_USERNAME = "username";
    private static final String PREF_STATUS = "status";
    private static final String PREF_PASSWORD = "password";

    // Setting activity's welcome message
    TextView welcome;

    // Instantiating a Database helper class
    private NotesDbAdapter mDbHelper;

    /*
     * (non-Javadoc)
     * @see android.app.Activity#onCreate(android.os.Bundle)
     * Called when the activity is first created.
     * This is the method that is called on start up to create what is
     * seen on the device.
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // setting the page layout from /res/layout/home.xml
        setContentView(R.layout.home);

        // Checking to see if this app has saved user preference for the user.
        SharedPreferences pref = getSharedPreferences(PREFS_NAME, MODE_PRIVATE);
        String username = pref.getString(PREF_USERNAME, null);

        if(username == null){
            username = "guest";
        }
        else{
            // Most all strings are saved in the /res/values/strings.xml file
            welcome = (TextView) this.findViewById(R.id.welcometext);
            welcome.setText(username);
        }

        // This will instantiate my database helper class, open the database connection and then call the
        // fillData() method.

        mDbHelper = new NotesDbAdapter(this);
    }
}

```

```

    mDbHelper.open();
    fillData();
}

/*
 * Called every time the activity is the active display on the device.
 * This will request all the note titles from the database and create
 * a display that is a selectable list
 */
private void fillData() {
    Cursor c = mDbHelper.fetchAllNotes();
    startManagingCursor(c);

    // A string array with all the note titles
    String[] from = new String[] { NotesDbAdapter.KEY_TITLE };

    // A interger array with the ids of the text, auto filled by the system/
    int[] to = new int[] { R.id.text1 };

    // Now create an adapter and set it to display using our row,
    // which is made from this page, the cursor of all the note titles,
    // from array and the to array. This is everything required to show a
    // list of selectable items on the display

    SimpleCursorAdapter notes =
        new SimpleCursorAdapter(this, R.layout.notes_row, c, from, to);
    setListAdapter(notes);
}

/*
 * (non-Javadoc)
 * @see android.app.Activity#onResume()
 * Called once the app is running and the user is returning to this activity,
 * making it the active display again.
 */
protected void onResume(){
    super.onResume();

    // Checking for user saved preferences
    SharedPreferences pref = getSharedPreferences(PREFS_NAME, MODE_PRIVATE);
    String username = pref.getString(PREF_USERNAME, null);

    // Setting the welcome message
    if(username == null){
        username = "guest";
    }
    welcome = (TextView) this.findViewById(R.id.welcometext);
}

```

```

welcome.setText(username);

// This will instantiate my database helper class, open the database connection and then call the
fillData() method.

mDbHelper = new NotesDbAdapter(this);
mDbHelper.open();
fillData();
}

/*
 * (non-Javadoc)
 * @see android.app.Activity#onOptionsItemSelected(android.view.Menu)
 * Creating the menu you see when you click the menu key on the device
 */
@Override
public boolean onOptionsItemSelected(Menu menu) {
    MenuInflater inflater = getMenuInflater();

    // This tells the system what the menu should look like and where to find it.
    // I have a menu layout saved under /res/menu/menu.xml
    inflater.inflate(R.menu.menu, menu);
    return true;
}

/*
 * (non-Javadoc)
 * @see android.app.Activity#onOptionsItemSelected(android.view.MenuItem)
 * Which menu button was selected and calling the correct method
 */
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.new_note:
            newNote();
            return true;
        case R.id.account:
            updateAccount();
            return true;
        case R.id.sync:
            syncNotes();
            return true;
        case R.id.policies:
            policies();
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}

```



```

}

/*
 * This method will start the policies activity
 */
private void policies() {
    // Intent is how I tell the system where it is and where I want it to go next
    Intent i = new Intent(HomeActivity.this, PoliciesActivity.class);

    // Tells the system to switch to this activity
    startActivity(i);
}

/*
 * This method checks what the user has saved for there getNote account information
 * and weather they are a current member, if so it will call the sync method, and if
 * not it will suggest to them to create an account
 */
private void syncNotes(){
    SharedPreferences pref = getSharedPreferences(PREFS_NAME, MODE_PRIVATE);
    String username = pref.getString(PREF_USERNAME, null);
    String status = pref.getString(PREF_STATUS, null);

    // Does the user have a valid getNote account
    boolean check = Boolean.parseBoolean(status);

    if (check == true){
        try {
            // Valid account then sync
            sync(username);
        }
        catch (Exception e) {
            e.printStackTrace();
        }
        catch (Throwable e) {
            e.printStackTrace();
        }
        // Refresh the page
        fillData();
    }
    else{
        // Not a valid account, they should create one
        Toast.makeText(HomeActivity.this, R.string.ToastCreate, Toast.LENGTH_LONG).show();
    }
}

/*

```

```

    * This method is my sync service. It will send all the notes on the phone to the server, which
    will update the
    * server with new or updated text notes, and then return the notes that the device need to save
    and or update
    */
private void sync(String username) throws Throwable, IOException {

    BufferedReader in = null;
try {
    mDbHelper = new NotesDbAdapter(this);
    mDbHelper.open();

    //getting all user notes out of database
    Cursor c = mDbHelper.fetchAllNotes();
    startManagingCursor(c);
    int totalNotes = c.getCount();
    String postNotes = "";

    if(totalNotes > 0)
    {
        c.moveToFirst();
        do{
            String id = c.getString(0);
            String title = c.getString(1);
            String body = c.getString(2);
            String date = c.getString(3);

            String file = title.substring(title.length() -3);

            //only want txt notes
            if(file.equals("txt")){
                //building json string (will do a php explode("[", this)
                postNotes = postNotes + date + "-" + username + "-" + title + "[" + body
                + "[";
            }
        }while(c.moveToNext());
    }

    int trim = postNotes.length();

    if(trim == 0 ){
        BufferedReader en = null;
        try {
            // Start set up to ask system to make a http call for us
            HttpClient client = new DefaultHttpClient();
            HttpGet request = new HttpGet();
            // Where the URL is that I want to call is at ( I wrote this php page as well for this project)

```

```

request.setURI(new URI("http://www.getnote.org/Android/json.php?user=" + username));
// Response is what is returned from the page, in this case a JSON string of notes
HttpResponse response = client.execute(request);
en = new BufferedReader
(new InputStreamReader(response.getEntity().getContent()));
StringBuffer sb = new StringBuffer("");
String line = "";
String NL = System.getProperty("line.separator");
while ((line = en.readLine()) != null) {
    sb.append(line + NL);
}
en.close();
String page = sb.toString();
// Now we have the JSON string break it apart in to individual notes and save them
JSONObject object = new JSONObject(page);
JSONArray note = object.names();
JSONArray value = object.toJSONArray(note);
for(int i = 0; i < value.length(); i++){
    String file = note.getString(i);
    String[] parts = file.split("-");
    String date = parts[0];
    String filename = parts[2]+"-"+parts[3]+"-"+parts[4]+"-"+parts[5];
    String submitNote = value.getString(i);

    long id = NotesDbAdapter.createNote(filename, submitNote, date);
}
}
finally {
    if (in != null) {
        try {
            in.close();
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }
}
}
else{
    postNotes = postNotes.substring(0, trim - 1);
    //creating json object to post to server
    JSONObject jsonObject = new JSONObject();
    jsonObject.put("sync", postNotes);

    // Start set up to ask system to make a http call for us
    HttpClient client = new DefaultHttpClient();
    HttpPost request = new HttpPost();

```

```

// Where the URL is that I want to call is at ( I wrote this php page as well for this
project)

request.setURI(new URI("http://www.getnote.org/Android/sync.php"));

//creating post object
List<NameValuePair> nVP = new ArrayList<NameValuePair>(2);
nVP.add(new BasicNameValuePair("json", jsonObject.toString()));

request.setEntity(new UrlEncodedFormEntity(nVP));
// Response is what is returned from the page, in this case a JSON string of notes
HttpResponse response = client.execute(request);
in = new BufferedReader
(new InputStreamReader(response.getEntity().getContent()));
StringBuffer sb = new StringBuffer("");
String line = "";
String NL = System.getProperty("line.separator");

while ((line = in.readLine()) != null) {
    sb.append(line + NL);
}

in.close();
//this is what is returned to be saved to device db
String page = sb.toString();
//Toast.makeText(HomeActivity.this, page , Toast.LENGTH_LONG).show();
// Now we have the JSON string break it apart in to individual notes and save them
JSONObject object = new JSONObject(page);
JSONArray note = object.names();
JSONArray value = object.toJSONArray(note);
for(int i = 0; i < value.length(); i++){
    String file = note.getString(i);
    String[] parts = file.split("-");
    String date = parts[0];
    String filename = parts[2]+"-"+parts[3]+"-"+parts[4]+"-"+parts[5];
    String submitNote = value.getString(i);

    if(!NotesDbAdapter.isTitleSaved(filename)){
        //false, this is a new note
        long id = NotesDbAdapter.createNote(filename, submitNote, date);
    }
    else
    { //true, this note needs to be updated, first to find its id
        //getting all user notes out of database
        Cursor e = mDbHelper.fetchAllNotes();
        startManagingCursor(e);
        int totalN = e.getCount();
    }
}

```

```

        String tNotes = "";

        if(totalNotes > 0)
        {
            e.moveToFirst();
            do{
                long id = e.getLong(0);
                String title = e.getString(1);

                if(title.equals(filename)){
                    boolean updated = NotesDbAdapter.updateNote(id,
                        filename, submitNote, date);
                }
            }while(e.moveToNext());
        }
    }
}

Toast.makeText(HomeActivity.this, R.string.ToastSync , Toast.LENGTH_LONG).show();
}
finally {
    if (in != null) {
        try {
            in.close();
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }
}
}

/*
 * This method is called with the user clicks the account menu button
 */
private void updateAccount() {
    // Intent is how I tell the system where it is and where I want it to go next
    Intent i = new Intent(HomeActivity.this, Login.class);

    // Tells the system to switch to this activity
    startActivity(i);
}

/*
 * This method is called when the user clicks the new note menu button
 */

```

```

private void newNote() {
    // Intent is how I tell the system where it is and where I want it to go next
    Intent i = new Intent(HomeActivity.this, SelectionActivity.class);

    // Tells the system to switch to this activity
    startActivity(i);
}

/*
 * (non-Javadoc)
 * @see android.app.ListActivity#onListItemClick(android.widget.ListView,
 * android.view.View, int, long)
 * This method is called when the user selects on of there notes from the selectable item list
 */
@Override
protected void onListItemClick(ListView l, View v, int position, long id) {
    super.onListItemClick(l, v, position, id);

    // Calls the noteSelected method with the id of the note that was selected
    noteSelected(id);
}

/*
 * This method is called once the user has selected a note from the selectable item list
 * and will determine what kind of note was selected and display it correctly for them
 * along with any other options such as view or delete for photo notes
 */
private void noteSelected(long id) {
    final long mId = id;
    // Call data base and return the rest of the information for that note
    Cursor c = NotesDbAdapter.fetchNote(id);
    int column = c.getColumnIndex("title");
    String filename = c.getString(column);
    int where = c.getColumnIndex("body");
    final String submitNote = c.getString(where);

    String file = filename.substring(filename.length() - 3);

    if(file.equals("txt")){
        // The note selected was a text note and we need to start the create note activity,
        // So the user can view, update or delete this note
        Intent i = new Intent(HomeActivity.this, CreateNoteActivity.class);
        i.putExtra(NotesDbAdapter.KEY_ROWID, id);
        startActivity(i);
    }
    else{
        // The note selected was a photo note, we need to ask if the user wants to view or delete this it
    }
}

```

```

// Alert Dialog will create and call a pop up asking the user if they want to view of delete this it
final AlertDialog alertDialog = new AlertDialog.Builder(HomeActivity.this).create();
alertDialog.setTitle(R.string.AlerDialogPhotoNote);
alertDialog.setMessage("please choose");
alertDialog.setButton("View", new DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int which) {
        // If they select view we ask the system to start the activity that will show a image
        Intent intent = new Intent();
        intent.setAction(Intent.ACTION_VIEW);
        intent.setDataAndType(Uri.parse(submitNote), "image/*");
        startActivity(intent);
    }
});
alertDialog.setButton2("Delete", new DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int which) {
        // They selected delete so close this dialog and open another asking if the are sure
        alertDialog.cancel();
        final AlertDialog alertDialog1 = new AlertDialog.Builder(HomeActivity.this).create();
        alertDialog1.setTitle(R.string.AlerDialogPhotoNote);
        alertDialog1.setMessage("are you sure you want to delete");
        alertDialog1.setButton("ok", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
                // They selected ok, call delete in the database class, and then inform the user it is
                // deleted

                boolean del = NotesDbAdapter.deleteNote(mId);
                Toast.makeText(HomeActivity.this, "photo has been deleted",
                    Toast.LENGTH_SHORT).show();
                // Refresh the screen
                fillData();
            }
        });
        alertDialog1.setButton2("cancel", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which)
            {
                // They selected cancel, close the dialog
                alertDialog1.cancel();
            }
        });
        // Display second dialog (are they sure)
        alertDialog1.show();
    }
});

// Display first dialog (what do they want to do)
alertDialog.show();

}
}

```

```
/*
 * (non-Javadoc)
 * @see android.view.View.OnClickListener#onClick(android.view.View)
 * This method must be here, I don't have a button that gets clicked, but the selectable item list
 * still requires it to be here
 */
@Override
public void onClick(View v) {
    // TODO Auto-generated method stub
}
}
```



## Login.java

```
package org.getnote;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.URI;
import java.net.URISyntaxException;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import org.json.JSONTokener;
import android.app.Activity;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class Login extends Activity implements OnClickListener {
    // These are the global variables, PREFS_ are used for saving and getting, users preference from
    // the system.
    // I save the users, getNote, user name, password, account status in the MyPrefsFile which is a
    // system file.
    public static final String PREFS_NAME = "MyPrefsFile";
    private static final String PREF_USERNAME = "username";
    private static final String PREF_PASSWORD = "password";
    private static final String PREF_STATUS = "status";
    private static boolean FIRST = false;

    EditText username;
    EditText password;
    Button submit;

    /*
     * (non-Javadoc)
     * @see android.app.Activity#onCreate(android.os.Bundle)
     * Called when the activity is first created.
     * This is the method that is called on start up to create what is
     * seen on the device.
    */
}
```

```

        */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.login);

    // Checking if user, already saved account information
    SharedPreferences pref = getSharedPreferences(PREFS_NAME, MODE_PRIVATE);
    String email = pref.getString(PREF_USERNAME, null);
    String pass = pref.getString(PREF_PASSWORD, null);
    String status = pref.getString(PREF_STATUS, null);

    if(email != null){
        username = (EditText) this.findViewById(R.id.editTextUsername);
        username.setText(email);

        if(status.equals("false")){
            FIRST = false;
        }
        else{
            FIRST = true;
        }
    }
    if(pass != null){
        password = (EditText) this.findViewById(R.id.editTextPassword);
        password.setText(pass);
    }

    submit = (Button) this.findViewById(R.id.buttonSubmitLogin);
    submit.setOnClickListener(this);
}

/*
 * (non-Javadoc)
 * @see android.view.View.OnClickListener#onClick(android.view.View)
 * This method is called when the user clicks the submit button
 */
@Override
public void onClick(View v) {
    username = (EditText) this.findViewById(R.id.editTextUsername);
    password = (EditText) this.findViewById(R.id.editTextPassword);
    String un = username.getText().toString();
    String pw = password.getText().toString();

    // User can't leave the username field empty
    if(un.equals("")){

```

```

        Toast.makeText(Login.this, "No username entered",
            Toast.LENGTH_LONG).show();
    }
    // User can't leave the password field empty
    else if (pw.equals("")){
        Toast.makeText(Login.this, "No password entered",
            Toast.LENGTH_LONG).show();
    }
    else{
        boolean check = false;

        // Call isUser to check if this account info is a valid getNote user
        try {
            check = isUser(un, pw);
        }
        catch (Exception e1) {
            e1.printStackTrace();
        }

        SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
        SharedPreferences.Editor e = settings.edit();

        e.putString("username", un);
        e.putString("password", pw);
        e.putString("status", Boolean.toString(check));
        e.commit();

        if (check == true && !FIRST){
            Toast.makeText(Login.this, "Account info saved - Valid User - Text Notes
            Downloaded", Toast.LENGTH_LONG).show();
            try {
                // They are a valid user, lets call getNote!
                getnote(un);
            }
            catch (Exception e1) {
                e1.printStackTrace();
            }
        }
        else if(FIRST){
            Toast.makeText(Login.this, R.string.ToastGoodAcc,
                Toast.LENGTH_LONG).show();
        }
        else{
            Toast.makeText(Login.this, R.string.ToastBadAcc,
                Toast.LENGTH_LONG).show();
        }
    }
    // This method call tells the system we are done with this activity, and to return to

```

```

        the previous one
        finish();
    }
}

/*
 * This method is what the application is all about, getting the users notes off the server and
 * saving them to
 * there device this only happens the first time you save an account on the device, as to prevent
 * a user having
 * all there friends sign in back to back, to store all their friends notes on there device. At this
 * time only
 * the users notes can be stored on the phone to prevent cheating.
 */
private void getnote(String un) throws URISyntaxException, ClientProtocolException,
IOException, JSONException {
    String username = un;
    BufferedReader in = null;
try {
    // Start set up to ask system to make a http call for us
    HttpClient client = new DefaultHttpClient();
    HttpGet request = new HttpGet();
    // Where the URL is that I want to call is at ( I wrote this php page as well for this project)
    request.setURI(new URI("http://www.getnote.org/Android/json.php?user=" + username));
    // Response is what is returned from the page, in this case a JSON string of notes
    HttpResponse response = client.execute(request);
    in = new BufferedReader
(new InputStreamReader(response.getEntity().getContent()));
    StringBuffer sb = new StringBuffer("");
    String line = "";
    String NL = System.getProperty("line.separator");
    while ((line = in.readLine()) != null) {
        sb.append(line + NL);
    }
    in.close();
    String page = sb.toString();
    // Now we have the JSON string break it apart in to individual notes and save them
    JSONObject object = new JSONObject(page);
    JSONArray note = object.names();
    JSONArray value = object.toJSONArray(note);
    for(int i = 0; i < value.length(); i++){
        String file = note.getString(i);
        String[] parts = file.split("-");
        String date = parts[0];
        String filename = parts[2]+"-"+parts[3]+"-"+parts[4]+"-"+parts[5];
        String submitNote = value.getString(i);

```

```

        long id = NotesDbAdapter.createNote(filename, submitNote, date);
    }
}
finally {
    if (in != null) {
        try {
            in.close();
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }
}

/*
 * This method is called to ask the server if the users account information is a valid getNote user
 */
private boolean isUser(String un, String pw) throws URISyntaxException, ClientProtocolException, IOException {
    String username = un;
    String password = pw;

    BufferedReader in = null;
    try {
        // Start set up to ask system to make a http call for us
        HttpClient client = new DefaultHttpClient();
        HttpGet request = new HttpGet();
        // Where the URL is that I want to call is at ( I wrote this php page as well for this project)
        request.setURI(new URI("http://www.getnote.org/Android/isuser.php?user=" + username +
            "&pass=" + password));
        // Response is what is returned from the page, in this case a JSON string of notes
        HttpResponse response = client.execute(request);
        in = new BufferedReader
            (new InputStreamReader(response.getEntity().getContent()));
        StringBuffer sb = new StringBuffer("");
        String line = "";
        String NL = System.getProperty("line.separator");

        while ((line = in.readLine()) != null) {
            sb.append(line + NL);
        }

        in.close();
        String page = sb.toString();

        try {

```

```

        // Now we have the JSON string break it apart to see if the user is a valid getNote user
        JSONObject object = (JSONObject) new
        JSONTokener(page).nextValue();
        String user = object.getString("user");
        boolean check = Boolean.parseBoolean(user);
        return check;
    } catch (JSONException e) {
        e.printStackTrace();
    }
}
finally {
    if (in != null) {
        try {
            in.close();
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }
}
return false;
}
}

```

## NotesDbAdapter.java

```
package org.getnote;

import java.sql.Date;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

public class NotesDbAdapter {
    // These are the global variables, KEY_s are used for each attribute I will be saving in the table

    public static final String KEY_DATE = "date";
    public static final String KEY_TITLE = "title";
    public static final String KEY_BODY = "body";
    public static final String KEY_ROWID = "_id";

    private static final String TAG = "NotesDbAdapter";
    private DatabaseHelper mDbHelper;
    private static SQLiteDatabase mDb;

    // If there is not already a getNote database on the device, this will be used to create one
    private static final String DATABASE_CREATE =
        "create table notes (_id integer primary key autoincrement, "
        + "title text not null, body text not null, date text not null);";

    // Naming the database, and the table
    private static final String DATABASE_NAME = "data";
    private static final String DATABASE_TABLE = "notes";
    private static final int DATABASE_VERSION = 2;

    private final Context mContext;

    /*
     * User the systems database helper class to create a database if needed
     */
    private static class DatabaseHelper extends SQLiteOpenHelper {

        DatabaseHelper(Context context) {
            super(context, DATABASE_NAME, null, DATABASE_VERSION);
        }

        /*
         * (non-Javadoc)

```

```

    * @see
android.database.sqlite.SQLiteOpenHelper#onCreate(android.database.sqlite.SQLiteDatabase)
    * systems create method for creating a new database
    */
    @Override
    public void onCreate(SQLiteDatabase db) {

        db.execSQL(DATABASE_CREATE);
    }

    /*
    * (non-Javadoc)
    * @see
android.database.sqlite.SQLiteOpenHelper#onUpgrade(android.database.sqlite.SQLiteDatabase, int,
int)
    * This is a system method for updating the database if the users device has updated it SQLite
version
    */
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        Log.w(TAG, "Upgrading database from version " + oldVersion + " to "
            + newVersion + ", which will destroy all old data");
        db.execSQL("DROP TABLE IF EXISTS notes");
        onCreate(db);
    }
}

/*
* This method is setting up database adapter, so we can use said database
*/
public NotesDbAdapter(Context ctx) {
    this.mCtx = ctx;
}

/*
* This method is calling open on the database
*/
public NotesDbAdapter open() throws SQLException {
    mDbHelper = new DatabaseHelper(mCtx);
    mDb = mDbHelper.getWritableDatabase();
    return this;
}

/*
* This method calls close on the database
*/
public void close() {

```



```

    mDbHelper.close();
}

/*
 * This method is the SQLite query for saving a new note
 */
public static long createNote(String title, String body, String date) {
    ContentValues initialValues = new ContentValues();
    initialValues.put(KEY_TITLE, title);
    initialValues.put(KEY_BODY, body);
    initialValues.put(KEY_DATE, date);

    return mDb.insert(DATABASE_TABLE, null, initialValues);
}

/*
 * This method is the SQLite query for deleting a note
 */
public static boolean deleteNote(long rowId) {

    return mDb.delete(DATABASE_TABLE, KEY_ROWID + "=" + rowId, null) > 0;
}

/*
 * This method is the SQLite query that will return all the notes in the database
 */
public Cursor fetchAllNotes() {

    return mDb.query(DATABASE_TABLE, new String[] {KEY_ROWID, KEY_TITLE,
        KEY_BODY, KEY_DATE}, null, null, null, null, null);
}

/*
 * This method is the SQLite query for returning a note, if the id is know
 */
public static Cursor fetchNote(long rowId) throws SQLException {

    Cursor mCursor =

        mDb.query(true, DATABASE_TABLE, new String[] {KEY_ROWID,
            KEY_TITLE, KEY_BODY, KEY_DATE}, KEY_ROWID + "=" + rowId, null,
            null, null, null, null);
    if (mCursor != null) {
        mCursor.moveToFirst();
    }
    return mCursor;
}

```

```

    }
    /*
    * This method is the SQLite query for updating an existing note
    */
    public static boolean updateNote(long rowId, String title, String body, String date) {
        ContentValues args = new ContentValues();
        args.put(KEY_TITLE, title);
        args.put(KEY_BODY, body);
        args.put(KEY_DATE, date);

        return mDb.update(DATABASE_TABLE, args, KEY_ROWID + "=" + rowId, null) > 0;
    }

    /*
    * This method is called to make sure a note's filename the user is trying to currently save is not
    stored
    */
    public static boolean isTitleSaved(String title){
        Cursor mCursor = null;
        try{
            mCursor = mDb.query(false, DATABASE_TABLE,
                                new String[]{KEY_TITLE}, KEY_TITLE + "=?",
                                new String[]{title}, null, null, null, null);

            //mCursor = mDb.

            while (mCursor.moveToNext()){
                return true;
            }
        }
        catch (SQLException sqle){
            Log.e(TAG, sqle.toString());
        }
        finally{
            if(mCursor != null && mCursor.isClosed()){
                mCursor.close();
            }
        }

        return false;
    }
}

```

## PoliciesActivity.java

```
package org.getnote;

import android.app.Activity;
import android.app.ListActivity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

public class PoliciesActivity extends Activity implements OnClickListener {
    Button back;
    TextView policies;
    TextView longP;

    /*
     * (non-Javadoc)
     * @see android.app.Activity#onCreate(android.os.Bundle)
     * Called when the activity is first created.
     * This is the method that is called on start up to create what is
     * seen on the device.
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.policies);
        back = (Button) this.findViewById(R.id.buttonback);
        back.setOnClickListener(this);
    }

    @Override
    public void onClick(View v) {
        // TODO Auto-generated method stub
        finish();
    }
}
```

## SelectionActivity.java

```
package org.getnote;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.util.Calendar;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.ContentValues;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.provider.MediaStore.Images.Media;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.Spinner;
import android.widget.Toast;

public class SelectionActivity extends Activity implements OnClickListener,
AdapterView.OnItemClickListener{

    // These are the global variables, PREFS_ are used for saving and getting, users preference from
    the system.
    // I save the users, getNote, user name, password, account status in the MyPrefsFile which is a
    system file.
    public static final String PREFS_NAME = "MyPrefsFile";
    private static final String PREF_FILENAME = "filename";

    private static final int CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE = 0;
    private static final String PREF_PASSWORD = "password";
    private String username = "";
    private String date = "";
    private String course = "";
    private String type = "";
    protected String filename="";
    private File values;
    private NotesDbAdapter mDbHelper;
```

```
String[] items= {"CSCI000 - Personal Notes",
    "CSCI101 - Introduction to Computer Science",
    "CSCI102 - Living With Technology",
    "CSCI111 - Programming and Algorithms I",
    "CSCI144 - Introduction to UNIX/Linux",
    "CSCI211 - Programming and Algorithms II",
    "CSCI221 - Assembly Language Programming",
    "CSCI301 - Computers Impact on Society",
    "CSCI311 - Algorithms and Data Structures",
    "CSCI313 - Mind in the Machine - Honors",
    "CSCI315 - Programming Languages",
    "CSCI317 - Linear Programming Applications",
    "CSCI320 - Computer Architecture",
    "CSCI340 - Operating Systems",
    "CSCI344 - Shell Programming",
    "CSCI346 - Introduction to Computer Networks and Network Mana",
    "CSCI351 - Numerical Methods Programming",
    "CSCI380 - Machines, Brains, and Minds",
    "CSCI381 - Language, Intelligence, and Computation",
    "CSCI389 - Industry Internship",
    "CSCI400 - Computer Science Activity",
    "CSCI430 - Software Engineering 1",
    "CSCI431 - Software Engineering 2",
    "CSCI444 - Fundamental UNIX System Administration",
    "CSCI465 - Web Programming Fundamentals",
    "CSCI490 - Directed Programming Experience",
    "CSCI498 - Topics in Computer Science",
    "CSCI499 - Honors Research Project/Thesis",
    "CSCI511 - Object-Oriented Programming",
    "CSCI515 - Compiler Design",
    "CSCI533 - Object-Oriented Analysis & Design",
    "CSCI540 - Systems Programming",
    "CSCI546 - Advanced Network Management",
    "CSCI547 - Advanced Computer Networks",
    "CSCI550 - Theory of Computing",
    "CSCI566 - Computer Graphics Programming",
    "CSCI567 - Graphical User Interfaces",
    "CSCI568 - Digital Image Processing",
    "CSCI569 - Advanced Computer Graphics",
    "CSCI580 - Artificial Intelligence",
    "CSCI583 - Expert Systems and Applications",
    "CSCI585 - Robotics and Machine Intelligence",
    "CSCI598 - Advanced Topics in Computer Science",
    "CSCI611 - Distributed Computing",
    "CSCI619 - Topics in Programming Language Theory",
    "CSCI620 - Computer Architecture",
    "CSCI629 - Topics in Computer Architecture",
```

```

"CSCI630 - Software Engineering - Grad",
"CSCI639 - Topics in Software Engineering - Grad",
"CSCI640 - Operating Systems - Grad",
"CSCI649 - Topics in Networking",
"CSCI650 - Design and Analysis of Algorithms",
"CSCI659 - Topics in Computer Theory",
"CSCI669 - Topics in Computer Graphics",
"CSCI679 - Topics in Database Systems",
"CSCI682 - Topics in Artificial Intelligence",
"CSCI693 - Research Methods in Computer Science",
"CSCI697 - Independent Study",
"CSCI698 - Seminar in Advanced Topics",
"CSCI699 - Masters Project",
"CSCI700 - Masters Thesis" };

```

```

String[] types = { "lecture",
                  "discussion",
                  "activity",
                  "lab",
                  "study_group",
                  "home_work" };

```

```

Button buttonTextNote;
Button buttonPixNote;
Spinner courseSpinner;
Spinner typeSpinner;

```

```

/*
 * (non-Javadoc)
 * @see android.app.Activity#onCreate(android.os.Bundle)
 * Called when the activity is first created.
 * This is the method that is called on start up to create what is
 * seen on the device.
 */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.input);
    //setting up the page spinner are the selection wheels for class and type
    courseSpinner = (Spinner) findViewById(R.id.course);
    ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this, R.array.courses,
android.R.layout.simple_spinner_item);
    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    courseSpinner.setOnItemSelectedListener(this);
    courseSpinner.setAdapter(adapter);
}

```

```

        typeSpinner = (Spinner) findViewById(R.id.type);
        ArrayAdapter<CharSequence> adaptor = ArrayAdapter.createFromResource(this, R.array.types,
        android.R.layout.simple_spinner_item);
        adaptor.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        typeSpinner.setOnItemSelectedListener(this);
        typeSpinner.setAdapter(adaptor);

        buttonTextNote = (Button)findViewById(R.id.buttonTextNote);
        buttonPixNote = (Button)findViewById(R.id.buttonPixNote);

        buttonTextNote.setOnClickListener(this);
        buttonPixNote.setOnClickListener(this);

    }

    /*
    * (non-Javadoc)
    * @see android.view.View.OnClickListener#onClick(android.view.View)
    * Called to find out what button the user clicked
    */
    @Override
    public void onClick(View arg0) {
        // mharris23@mail.csuchico.edu-2011.9.26-CSCI490-lecture-.txt
        // user-date-course-type-.file

        // Checking to see if this app has saved user preference for the user.
        SharedPreferences pref = getSharedPreferences(PREFS_NAME, MODE_PRIVATE);

        SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
        SharedPreferences.Editor e = settings.edit();

        e.putString("filename", filename);
        e.commit();

        DatePicker lecdate = (DatePicker)findViewById(R.id.lectureDate);
        String day = Integer.toString(lecdate.getDayOfMonth());
        String month = Integer.toString(lecdate.getMonth() + 1);
        String year = Integer.toString(lecdate.getYear());
        date = year+"."+month+"."+day;
        filename = date+"-"+course+"-"+type;
        // filename = username+"-"+date+"-"+course+"-"+type+"-.txt";

        switch (arg0.getId()){
        case R.id.buttonTextNote:

```

```

        // They clicked text note
        takeTextNote(filename);
    break;
    case R.id.buttonPixNote:
        // They clicked photo note pop up dialog asking camera or images
        final AlertDialog alertDialog = new
AlertDialog.Builder(SelectionActivity.this).create();
        alertDialog.setTitle(R.string.AlerDialogPhotoNote);
        alertDialog.setMessage("add photo");
        alertDialog.setButton("camera", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
                // They clicked camera
                takePhotoNote(filename);
            }
        });
        alertDialog.setButton2("images", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
                // They clicked images
                getPhotoNote(filename);
            }
        });

        alertDialog.show();
        break;
    }
}

/*
 * This method ask the system what apps can show the user their images and to start that
activity
 */
private void getPhotoNote(String filename){
    filename = filename + ".jpg";
    if(!NotesDbAdapter.isTitleSaved(filename)){
        // false= then we can save this note, the filename is unique
        SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
        SharedPreferences.Editor e = settings.edit();

        e.putString("filename", filename);
        e.commit();

        Intent intent = new Intent();
        intent.setType("image/*");
        intent.setAction(Intent.ACTION_GET_CONTENT);

        // Setting up my return code as 17
        startActivityForResult(Intent.createChooser(intent,
        "Select Picture"), 17);
    }
}

```



```

        else{
            // true=then this note is in DB already
            Toast.makeText(SelectionActivity.this, R.string.ToastNotUnique,
Toast.LENGTH_LONG).show();
        }
    }

    /*
    * This method ask the system what apps can take pictures and to start that activity
    */
    private void takePhotoNote(String filename){
        filename = filename + "-.jpg";
        if(!NotesDbAdapter.isTitleSaved(filename)){
            // false= then we can save this note, the filename is unique

            File f = new File(Environment.getExternalStorageDirectory() + "/getNote");

            if(!f.exists()){
                Toast.makeText(this, f.toString(), Toast.LENGTH_SHORT).show();
                f.mkdir();
            }

            values = new File (Environment.getExternalStorageDirectory() + "/getNote/" +
filename);

            SharedPreferences settings = getSharedPreferences(PREFS_NAME, 0);
            SharedPreferences.Editor e = settings.edit();

            e.putString("filename", filename);
            e.commit();

            Intent cam = new Intent(MediaStore.ACTION_IMAGE_CAPTURE );

            // Setting up my return code as 15
            cam.putExtra( MediaStore.EXTRA_OUTPUT, Uri.fromFile(values) );
            startActivityForResult(cam, 15);
        }
        else{
            // true=then this note is in DB already
            Toast.makeText(SelectionActivity.this, R.string.ToastNotUnique,
Toast.LENGTH_LONG).show();
        }
    }

    /*
    * This method will call the CreateNoteActivity so the user can take a new note
    */

```

```

private void takeTextNote(String filename) {
    filename = filename + ".txt";
    if(!NotesDbAdapter.isTitleSaved(filename)){
        // false= then we can save this note, the filename is unique

        Intent i = new Intent(SelectionActivity.this, CreateNoteActivity.class);
        Bundle b = new Bundle();
        b.putString("filename", filename);
        i.putExtras(b);
        startActivity(i);
        finish();
    }
    else{
        // true=then this note is in DB already
        Toast.makeText(SelectionActivity.this, R.string.ToastNotUnique,
            Toast.LENGTH_LONG).show();
    }
}

}

/*
 * (non-Javadoc)
 * @see android.app.Activity#onActivityResult(int, int, android.content.Intent)
 * This method is called when returning from the systems camera or image activity
 */
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    int result = resultCode;
    int request = requestCode;
    String submitNote = "";

    SharedPreferences pref = getSharedPreferences(PREFS_NAME, MODE_PRIVATE);
    String filename = pref.getString(PREF_FILENAME, null);

    //15 was the request code for returning from the system camera activity
    if (request == 15){
        if (result == -1){

            File path = new File (Environment.getExternalStorageDirectory() +
                "/getNote/" + filename);
            submitNote = "file://" + path.toString();
        }
        else{
            Toast.makeText(SelectionActivity.this, R.string.ToastPhotoNotSaved ,
                Toast.LENGTH_LONG).show();
        }
    }
    // 17 was the request code from returing from the system image activity

```

```

        else if (request == 17){
            if(result == -1){
                Uri choose = data.getData();
                submitNote = choose.toString();
            }
            else{
                Toast.makeText(SelectionActivity.this, R.string.ToastPhotoNotSaved ,
                    Toast.LENGTH_LONG).show();
            }
        }
        if(result == -1){
            mDbHelper = new NotesDbAdapter(this);
            mDbHelper.open();
            String date = Integer.toString(Calendar.DATE);
            long id = NotesDbAdapter.createNote(filename, submitNote, date);
            Toast.makeText(SelectionActivity.this, R.string.ToastPhotoNoteSaved,
                Toast.LENGTH_SHORT).show();
        }
        finish();
    }
    /*
    * (non-Javadoc)
    * @see
    android.widget.AdapterView.OnItemClickListener#onItemSelected(android.widget.AdapterView,
    android.view.View, int, long)
    * This method is called when the user selects a class and type
    */
    @Override
    public void onItemSelected(AdapterView<?> arg0, View arg1, int arg2,
        long arg3) {
        switch (arg0.getId()){
            case R.id.course:
                course = items[arg2];
                course=course.substring(0, course.indexOf(" "));
                break;
            case R.id.type:
                type = types[arg2];
                break;
        }
    }
    /*
    * (non-Javadoc)
    * @see
    android.widget.AdapterView.OnItemClickListeneronNothingSelected(android.widget.AdapterView)
    * This is called if the user does not selected anything which implies they wanted the first thing in list
    */
    @Override

```

```

public void onNothingSelected(AdapterView<?> arg0) {
    switch (arg0.getId()){
    case R.id.course:
        course = items[0];
        //Changed this was
        //course=course
        //????
        course=course.substring(0, course.indexOf(" "));
        break;
    case R.id.type:
        type = types[0];
        break;
    }
}
}

```

## AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the AndroidManifest.xml, This is how the .xml files get parsed correctly, -->
<!-- also where all the system permissions are kept -->
<!-- It was created in Eclipse using the Android SDK tools -->
<!-- A interactive GUI for creating layouts -->
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="org.getnote" android:versionCode="1" android:versionName="1.0">
    <uses-sdk android:minSdkVersion="4" />
    <uses-permission android:name="android.permission.CAMERA"></uses-permission>
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"></uses-permission>
    <uses-permission android:name="android.permission.INTERNET"></uses-permission>

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".HomeActivity" android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="SelectionActivity"></activity>
        <activity android:name="CreateNoteActivity"></activity>
        <activity android:name="Login"></activity>
        <activity android:name="PoliciesActivity"></activity>

    </application>
</manifest>
```

## layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the page layout for the HomeActivity.java, -->
<!-- It was created in Eclipse using the Android SDK tools -->
<!-- A interactive GUI for creating layouts -->
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent" android:background="@drawable/img01">
    <TextView android:layout_height="wrap_content" android:id="@+id/textView1"
        android:textAppearance="?android:attr/textAppearanceLarge" android:layout_width="fill_parent"
        android:textColor="#E44D32" android:text="@string/titleHome" android:gravity="center"
        android:textSize="28dp"></TextView>
    <TextView android:layout_height="wrap_content" android:textAppearance="?
        android:attr/textAppearanceSmall" android:id="@+id/welcometext" android:textColor="#4D8D99"
        android:gravity="center" android:layout_width="fill_parent" android:textSize="18dp"></TextView>
    <ListView android:id="@+id/android:id/list"
        android:layout_height="wrap_content" android:layout_width="fill_parent"/>
    <TextView android:id="@+id/android:id/empty"
        android:layout_height="wrap_content"
        android:text="@string/none" android:textColor="#E44D32" android:layout_width="fill_parent"
        android:textSize="22dp" android:gravity="center"/>

</LinearLayout>
```

## input.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the page layout for the SelectionActivity.java, -->
<!-- It was created in Eclipse using the Android SDK tools -->
<!-- A interactive GUI for creating layouts -->
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" android:orientation="vertical"
    android:background="@drawable/img01">
    <TextView android:layout_height="wrap_content" android:id="@+id/textView1"
    android:textAppearance="?android:attr/textAppearanceLarge" android:layout_width="fill_parent"
    android:textColor="#E44D32" android:text="@string/titleSelection" android:gravity="center"
    android:textSize="28dp"></TextView>
    <Spinner android:layout_width="match_parent" android:layout_height="wrap_content"
    android:id="@+id/course" android:prompt="@string/promptClassInput"></Spinner>
    <Spinner android:layout_width="match_parent" android:layout_height="wrap_content"
    android:id="@+id/type" android:prompt="@string/promptType"></Spinner>
    <DatePicker android:layout_width="fill_parent" android:id="@+id/lectureDate"
    android:layout_height="wrap_content"></DatePicker>
    <TextView android:id="@+id/textView2" android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall" android:layout_width="fill_parent"
    android:layout_weight="1"></TextView>
    <LinearLayout android:layout_height="wrap_content" android:layout_width="match_parent"
    android:id="@+id/linearLayout1">
        <Button android:layout_gravity="center" android:layout_height="wrap_content"
        android:text="@string/buttonTextNote" android:id="@+id/buttonTextNote"
        android:layout_width="wrap_content" android:layout_weight="50"></Button>
        <Button android:layout_height="wrap_content" android:text="@string/buttonPixNote"
        android:id="@+id/buttonPixNote" android:layout_width="wrap_content"
        android:layout_weight="50"></Button>
    </LinearLayout>
</LinearLayout>
```

## login.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the page layout for the Login.java, -->
<!-- It was created in Eclipse using the Android SDK tools -->
<!-- A interactive GUI for creating layouts -->
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent" android:background="@drawable/img01">
    <TextView android:id="@+id/textView1" android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceLarge" android:text="@string/textUserName"
    android:textColor="#E44D32" android:layout_width="fill_parent" android:gravity="center"
    android:textSize="28dp"></TextView>
    <EditText android:layout_height="wrap_content" android:layout_width="match_parent"
    android:id="@+id/editTextUsername">
        <requestFocus></requestFocus>
    </EditText>
    <TextView android:id="@+id/textView2" android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceLarge" android:text="@string/textPassword"
    android:textColor="#E44D32" android:layout_width="fill_parent" android:gravity="center"
    android:textSize="28dp"></TextView>
    <EditText android:layout_height="wrap_content" android:layout_width="match_parent"
    android:inputType="textPassword" android:id="@+id/editTextPassword"></EditText>
    <TextView android:id="@+id/textView3" android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceLarge" android:layout_width="wrap_content"
    android:layout_weight="1"></TextView>
    <Button android:layout_height="wrap_content" android:layout_width="fill_parent"
    android:text="@string/buttonsubmit" android:id="@+id/buttonSubmitLogin"></Button>

</LinearLayout>
```



#### notes\_row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the page layout for the HomeActivity.java/rows, -->
<!-- It was created in Eclipse using the Android SKD tools -->
<!-- A interactive GUI for creating layouts -->
<TextView android:id="@+id/text1" xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_height="wrap_content" android:textColor="#000000"
    android:layout_width="fill_parent" android:gravity="center" android:textSize="20dp"/>
```

#### note\_policies.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the page layout for the PoliciesActivity/rows, -->
<!-- It was created in Eclipse using the Android SKD tools -->
<!-- A interactive GUI for creating layouts -->
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent" android:background="@drawable/img01"
    android:isScrollContainer="true" android:overScrollMode="always" android:scrollbars="vertical">
    <TextView android:textAppearance="?android:attr/textAppearanceLarge"
    android:layout_height="wrap_content" android:gravity="center" android:layout_width="fill_parent"
    android:textSize="28dp" android:textColor="#E44D32" android:text="@string/TextPolicies"
    android:id="@+id/policies"></TextView>
    <TextView android:textAppearance="?android:attr/textAppearanceMedium"
    android:layout_width="fill_parent" android:text="@string/TextFullPolicies"
    android:layout_height="wrap_content" android:id="@+id/longP" android:textColor="#000000"
    android:textSize="15dp" android:isScrollContainer="true" android:layout_weight="50"></TextView>
    <Button android:text="Back" android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:id="@+id/buttonback"></Button>

</LinearLayout>
```

### submit.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the page layout for the CreateNoteActivity.java, -->
<!-- It was created in Eclipse using the Android SDK tools -->
<!-- A interactive GUI for creating layouts -->
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="fill_parent"
    android:layout_height="fill_parent" android:background="@drawable/img01">
    <TextView android:layout_width="fill_parent"
        android:layout_height="wrap_content" android:cursorVisible="false"
        android:password="false" android:gravity="center"
        android:text="@string/titleSubmitANote"
        android:textColor="#E44D32" android:textSize="28dp"/>
    <TextView android:layout_height="wrap_content" android:textAppearance="?
        android:attr/textAppearanceSmall" android:gravity="center" android:layout_width="fill_parent"
        android:textColor="#4D8D99" android:id="@+id/textViewFilename"
        android:textSize="18dp"></TextView>
    <EditText android:inputType="textMultiLine"
        android:layout_width="fill_parent" android:layout_height="fill_parent"
        android:id="@+id/editSubmitNote" android:hint="@string/hintFileName"
        android:layout_weight="1">
        <requestFocus></requestFocus>
    </EditText>
    <LinearLayout android:layout_height="wrap_content" android:id="@+id/linearLayout1"
        android:layout_width="match_parent">
        <Button android:layout_height="wrap_content"
            android:text="@string/buttonSubmiteNote" android:id="@+id/buttonSubmitNote"
            android:layout_width="wrap_content" android:layout_weight="50"
            android:layout_gravity="center"></Button>
        <Button android:layout_width="wrap_content" android:layout_height="wrap_content"
            android:layout_weight="50" android:layout_gravity="center"
            android:text="@string/buttonDeleteNote" android:id="@+id/buttonDeleteNote"></Button>
    </LinearLayout>
</LinearLayout>
```

### text1.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the page layout for the HomeActivity.java/selectable items list, -->
<!-- It was created in Eclipse using the Android SKD tools -->
<!-- A interactive GUI for creating layouts -->
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView android:id="@+id/text1"
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:textColor="#000000"/>

</LinearLayout>
```

### menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the page layout for the HomeActivity.java Menu, -->
<!-- It was created in Eclipse using the Android SKD tools -->
<!-- A interactive GUI for creating menus -->
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/new_note"
        android:title="@string/newNote" android:icon="@drawable/newnote"/>

    <item android:id="@+id/sync" android:icon="@drawable/sync"
        android:title="@string/menusync"></item><item android:title="@string/menuaccount"
        android:id="@+id/account" android:icon="@drawable/account"></item>
    <item android:id="@+id/policies" android:title="@string/menuPolicies"
        android:icon="@drawable/policies"></item>
</menu>
```

## strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- This is the strings.xml which holds all the key value pairs -->
<!-- It was created in Eclipse using the Android SDK tools -->
<!-- A interactive GUI for saving strings -->
<resources>
    <string name="titleSubmitANote">submit a text note</string>
    <string name="app_name">getNote</string>
    <string name="buttonSubmiteNote">Submit Note</string>
    <string name="hintFileName">Note</string>
    <string name="promptClassInput">Course</string>
    <string name="buttonTextNote">Text Note</string>
    <string name="buttonPixNote">Photo Note</string>
    <string name="titleSelection">class / type / date</string>
    <string name="titleHome">welcome to getNote</string>
    <string name="newNote">new note</string>
    <string name="promptType">Course Note Type</string>
    <string-array name="types">
        <item>lecture</item>
        <item>discussion</item>
        <item>activity</item>
        <item>lab</item>
        <item>study_group</item>
        <item>home_work</item>
    </string-array>
    <string-array name="courses">
        <item>CSCI000 - Personal Notes</item>
        <item>CSCII101 - Introduction to Computer Science</item>
        <item>CSCII102 - Living With Technology</item>
        <item>CSCII111 - Programming and Algorithms I</item>
        <item>CSCII144 - Introduction to UNIX/Linux</item>
        <item>CSCII211 - Programming and Algorithms II</item>
        <item>CSCII221 - Assembly Language Programming</item>
        <item>CSCII301 - Computers Impact on Society</item>
        <item>CSCII311 - Algorithms and Data Structures</item>
        <item>CSCII313 - Mind in the Machine - Honors</item>
        <item>CSCII315 - Programming Languages</item>
        <item>CSCII317 - Linear Programming Applications</item>
        <item>CSCII320 - Computer Architecture</item>
        <item>CSCII340 - Operating Systems</item>
        <item>CSCII344 - Shell Programming</item>
        <item>CSCII346 - Introduction to Computer Networks</item>
        <item>CSCII351 - Numerical Methods Programming</item>
        <item>CSCII380 - Machines, Brains, and Minds</item>
        <item>CSCII381 - Language, Intelligence, and Computation</item>
        <item>CSCII389 - Industry Internship</item>
        <item>CSCII400 - Computer Science Activity</item>
    </string-array>
</resources>
```

- <item>CSCI430 - Software Engineering 1</item>
- <item>CSCI431 - Software Engineering 2</item>
- <item>CSCI444 - Fundamental UNIX System Administration</item>
- <item>CSCI465 - Web Programming Fundamentals</item>
- <item>CSCI490 - Directed Programming Experience</item>
- <item>CSCI498 - Topics in Computer Science</item>
- <item>CSCI499 - Honors Research Project/Thesis</item>
- <item>CSCI511 - Object-Oriented Programming</item>
- <item>CSCI515 - Compiler Design</item>
- <item>CSCI533 - Object-Oriented Analysis and Design</item>
- <item>CSCI540 - Systems Programming</item>
- <item>CSCI546 - Advanced Network Management</item>
- <item>CSCI547 - Advanced Computer Networks</item>
- <item>CSCI550 - Theory of Computing</item>
- <item>CSCI566 - Computer Graphics Programming</item>
- <item>CSCI567 - Graphical User Interfaces</item>
- <item>CSCI568 - Digital Image Processing</item>
- <item>CSCI569 - Advanced Computer Graphics</item>
- <item>CSCI580 - Artificial Intelligence</item>
- <item>CSCI583 - Expert Systems and Applications</item>
- <item>CSCI585 - Robotics and Machine Intelligence</item>
- <item>CSCI598 - Advanced Topics in Computer Science</item>
- <item>CSCI611 - Distributed Computing</item>
- <item>CSCI619 - Topics in Programming Language Theory</item>
- <item>CSCI620 - Computer Architecture</item>
- <item>CSCI629 - Topics in Computer Architecture</item>
- <item>CSCI630 - Software Engineering - Grad</item>
- <item>CSCI639 - Topics in Software Engineering - Grad</item>
- <item>CSCI640 - Operating Systems - Grad</item>
- <item>CSCI649 - Topics in Networking</item>
- <item>CSCI650 - Design and Analysis of Algorithms</item>
- <item>CSCI659 - Topics in Computer Theory</item>
- <item>CSCI669 - Topics in Computer Graphics</item>
- <item>CSCI679 - Topics in Database Systems</item>
- <item>CSCI682 - Topics in Artificial Intelligence</item>
- <item>CSCI693 - Research Methods in Computer Science</item>
- <item>CSCI697 - Independent Study</item>
- <item>CSCI698 - Seminar in Advanced Topics</item>
- <item>CSCI699 - Masters Project</item>
- <item>CSCI700 - Masters Thesis</item>

</string-array>

<string name="PREF\_USERNAME">username</string>

<string name="PREF\_NAME">MyPrefsFile</string>

<string name="PREF\_PASSWORD">password</string>

<string name="buttonsubmit">Submit</string>

<string name="textUserName">School Email</string>

<string name="textPassword">getNote.org Password</string>

```

<string name="menuaccount">account</string>
<string name="none">you have no notes</string>
<string name="buttonDeleteNote">Delete</string>
<string name="ToastAccountInfoReq">Account Informatoin Required</string>
<string name="ToastNotUnique">Note with this class / type / date exist. Try Changing the
type</string>
<string name="ToastEmptyNote">You can not save a empty note</string>
<string name="ToastPhotoNotSaved">Your note was not saved</string>
<string name="AlerDialogPhotoNote">getNote</string>
<string name="AlertDialogText">please choose</string>
<string name="ToastPhotoNoteSaved">Photo Note has been saved</string>
<string name="EditTextReEnterPassword">Re-enter Password</string>
<string name="menusync">sync</string>
<string name="ToastSaved">Note has been saved</string>
<string name="ToastUpdated">Note has been updated</string>
<string name="DialogDelete">are you sure you want to delete</string>
<string name="ToastDeleted">Note has been deleted</string>
<string name="ToastNot">Note was not saved</string>
<string name="ToastCreate">Please create account at www.getnote.org</string>
<string name="ToastGoodAcc">Account info saved</string>
<string name="ToastBadAcc">Account info saved - Not a Valid User</string>
<string name="menuPolicies">policies</string>
<string name="TextPolicies">Policies</string>
<string name="TextFullPolicies">getNote has every intention of being in compliance with the
Academic Integrity policy of your school. We are not, however, liable for any action taken by your
school against you the student for the use of this application. We suggest to all our members that they
familiarize themselves with the policies of their institution. This is a application for taking and saving
notes created by its users. To that end, we require that any documentation posted to and from this
application be original material created by the submitter. This application is for notes and notes only.
We reserve the right to edit or remove any content deemed outside the scope of this site, not limited to:
test or quiz material, assignments in any form, or copyrighted material (without citation). We value the
privacy of our users, as well as non-members. We will not collect any information from our users, and
will not share their school email address with secondary sources. For this same reason we will not
allow any personal information to be stored in notes uploaded to this site. We reserve the right to
remove such material. This application is not affiliated with CSU, Chico or any other school.
Membership on this site is a privilege - not a students right. We reserve the right to ban users from the
getNote site (both temporarily, and indefinitely) for violating any of the aforementioned policies on this
page.</string>
<string name="ToastSync">Your accounts have been synced</string>
</resources>

```