**Invisino**

**Part 1: Requirements Documentation**

**CS 495**

**Submitted by:**

**Maggie Gembala**

**Nicole Tollman**

**Jessie Melton**

**Justin Estep**

**The University of Alabama**

**Tuscaloosa, Alabama 35487**

**October 3, 2017**

**Update History:**

* Creation, Set up outline – September 19, 2017 Maggie Gembala
* Added activity diagrams – September 24, 2017 Maggie Gembala
* Added use case diagrams – September 25, 2017 Maggie Gembala

**Table of Contents:**

Section Page Number

1. [Update History](#Update_History)
2. [Table of Contents](#Table_of_Contents)
3. [Introduction](#Introduction)
   1. [Motivation](#Motivation)
   2. [Scope](#Scope)
   3. [Goals](#Goals)
   4. [Relevant Definitions](#Relevant_Definitions)
4. [Project Description](#Project_Description)
   1. [Primary Features](#Primary_Features)
   2. [Secondary Features](#Secondary_Features)
5. [Summary of Requirements](#Summary_of_Requirements)
   1. [Functional Requirements](#Functional_Requirements)
   2. [Nonfunctional Requirements](#Nonfunctional_Requirements)
6. [Diagrams](#Diagrams)
   1. [Use Case Diagrams](#Use_Case_Diagrams)
   2. [Activity Diagrams](#Activity_Diagrams)
   3. [Class Diagrams](#Class_Diagrams)
      1. [High Level](#High_Level_Class_Diagram)
      2. [Detailed](#Detailed_Class_Diagram)
   4. [Sequence Diagrams](#Sequence_Diagrams)

**I Introduction:**

1. **Motivation**

This application’s motivation started with the want to provide a more contemporary and renewed method of communication. With the use of augmented reality, our project brings connection into a new realm. The virtual world itself allows people to experience a fresh outlook on their surroundings, and combining this fact with communication opens new doors for people to reach out to others. By developing virtual notes overlaying the real-time camera feed, a unique environment for sharing ideas is formed. Tagged to a geographical location and timed for a specific period, these notes allow users to converse anonymously and explore their ideas freely using AR technology; this empowers people's creativity. The app offers varying uses ranging from interactive message passing to exciting scavenger hunts - the possibilities are endless.

1. **Scope**

The scope of this project is extremely scalable. Since the notes are both geographically tied to a location and live only for the specified time period, Invisino can be used both locally and globally. While the notes right now are all public, there is the possibility of branching out to having different groups and possibility of having hidden notes and conversations. As for right now, this is a stretch goal but is definitely a possibility for this application in the future.

1. **Goals**
2. **Relevant Definitions**

**II Project Description:**

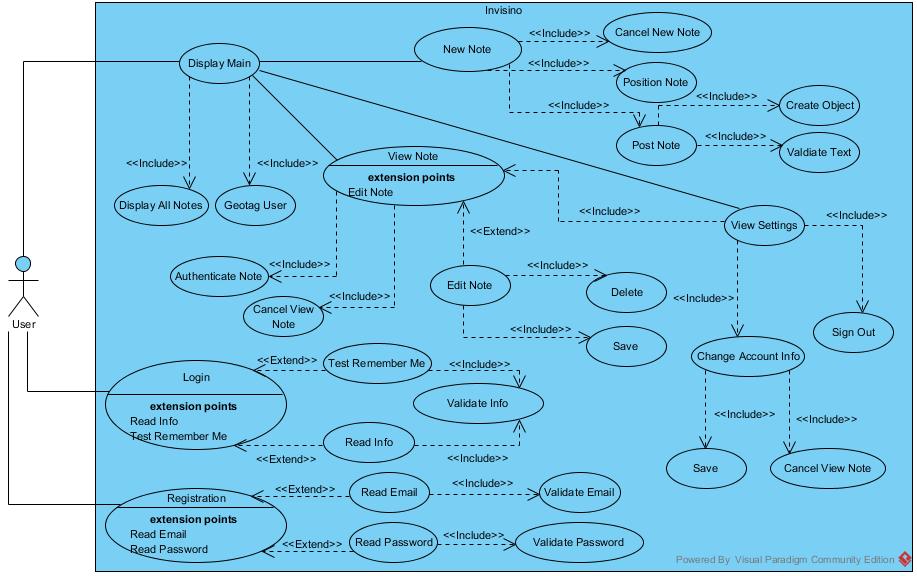
1. **Primary Features**
2. **Secondary Features**

**III Summary of Requirements:**

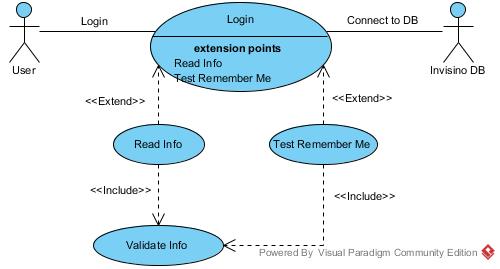
1. **Functional Requirements**
2. **Nonfunctional Requirements**

**IV Diagrams:**

1. **Use Case Diagrams**



The diagram above is the overall use case diagram, and the individual ones follow below. These describe all of the different possible routes that a user of Invisino can go through. The main three options are to login, register, and display the main screen once the user has signed in with their account. From the main screen, there are several more subsequent options that include viewing existent notes within range, creating a new note, and viewing one’s settings. If a user views a note that they made, they will be able to edit the text within the note or delete it altogether. If the user goes to their settings, they have the ability to view (and edit) all of their current notes, change their account information, or simply sign out. The descriptions of each use case starts here:



**Use Case:** *Login*

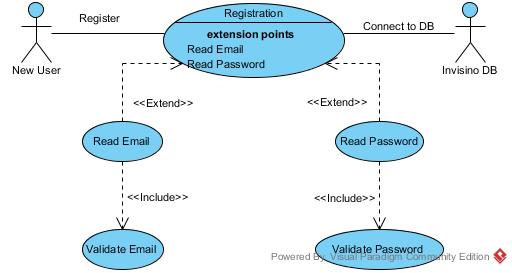
**Context:** The “Login” user case will allow a user to login with their email and password in order to access the rest of the application. If they have already logged in before and checked the Remember Me flag, the app will automatically login for them.

**Actors:** General User

**Main Success Story:**

1. Invisino DB checks if given device has already checked Remember Me flag for a given user, and login for them if it is true.

2. If Remember Me is false, the user will have to login using their credentials. The Invisino DB will authenticate their info and login if they are correct, otherwise prompt user to re-enter their info.



**Use Case:** *Registration*

**Context:** The “Registration” use case allows a new user to sign up an account to use the app. They will have to enter their email and create a password to do so.

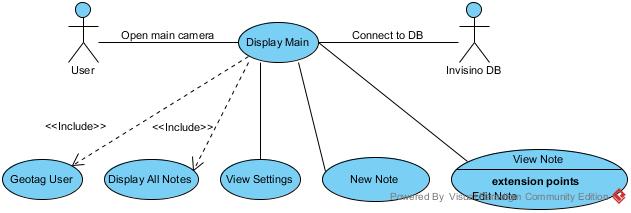
**Actors:** New User

**Main Success Story:**

1. Invisino DB will read the email and password that the new user enters and validate if they are acceptable.

2. If acceptable info has been entered, the user will be redirected to the login page and be logged in.

3. If the info is not acceptable, then they will be prompted to re-enter their info according to the requirements.



**Use Case:** *Display Main*

**Context:** The “Display Main” user case follows immediately after a user has logged in and is the main screen of the application. Here, the user is shown all notes within range to their physical location and has the option to view these notes, create their own note(s), or view their settings.

**Actors:** General User

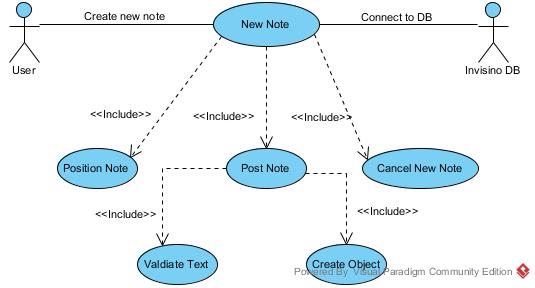
**Main Success Story:**

1. Invisino DB geotags the user’s device and pulls all of the nearby notes in range to display to the user.

2. If the user taps on the new note button, the “New Note” use case starts.

3. If the user taps on an existing note, the “View Note” use case starts.

4. If the user taps on the settings button, the “View Settings” use case starts.



**Use Case:** *New Note*

**Context:** The “New Note” use case allows a user to create their own note that will be geotagged to their current location. They

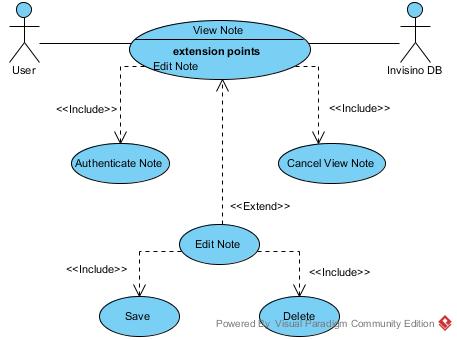
**Actors:** General User

**Main Success Story:**

1. The user will position the note before they add text to it.

2. Once positioned, the user will be prompted to provide text for the note. If they press on the “Post” button, the text will be validated; if the text is ok, then the note will be successfully posted, otherwise the user will be prompted to change their text to match the requirements. Once valid text is entered, Invisino DB will create that note object.

3. If at any point a user wants to cancel making a new note, they simply press cancel and will be returned to the main screen.



**Use Case:** *View Note* (and within, *Edit Note*)

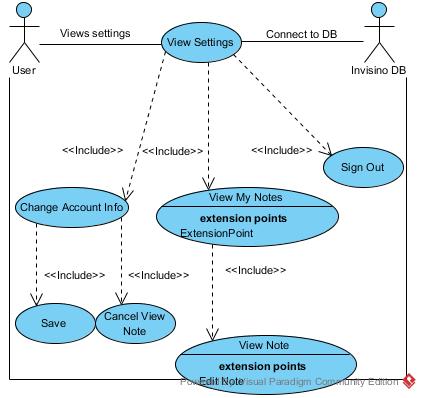
**Context:** The “View Note” user case allows a user to look at an already existing note. If the user is the creator of that given note, then they will be given the option to edit it.

**Actors:** General User

**Main Success Story:**

1. Invisino DB will authenticate if the user is the one made the note. If so, they will be able to open the sub use case “Edit Note”. Within that, they can either save their changes or delete the note entirely.

2. If at any time the user wants to go back to the main screen, they can cancel the view.



**Use Case:** *View Settings*

**Context:** The “View Settings” use case will allow a user to change their account info, view their current notes, or simply sign out. There is always a back button to go back to the main screen as well.

**Actors:** General User

**Main Success Story:**

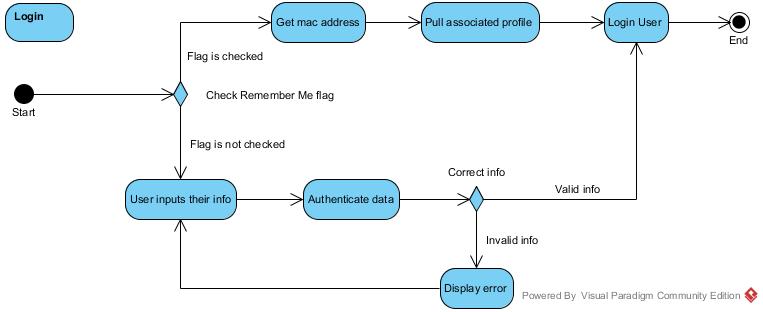
1. If the user wants to change their account info, they can edit their current email and password, so long as the new info is valid.

2. If the user wants to view their notes, they will be shown a list of their current notes. They can view and edit each one of them as wanted.

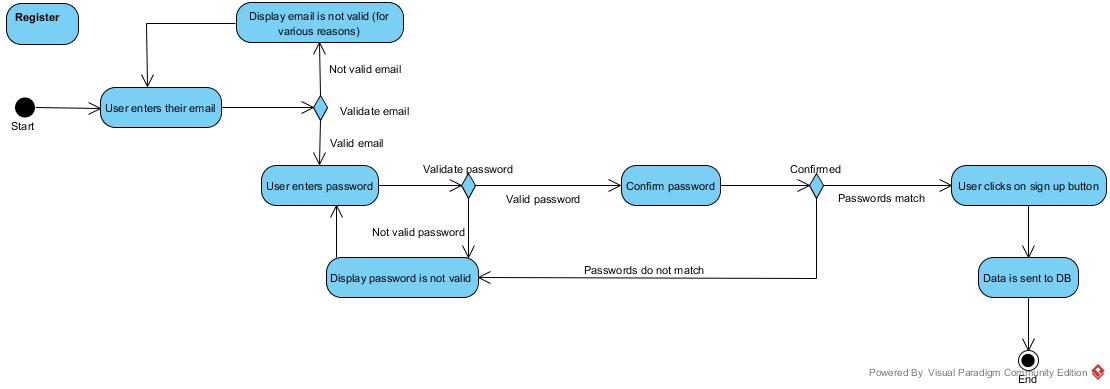
3. The user can press the sign out button if they want to switch accounts or get out of theirs for any reason.

1. **Activity Diagrams**

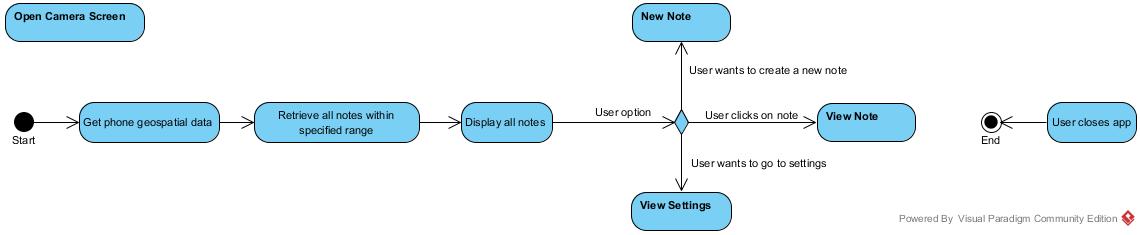
These diagrams demonstrate the flow of actions that can be taken when a user opens and uses the application. At any point, the user can close the application, ending the currently running activity. When Invisino opens, the first activity is the login activity.

****

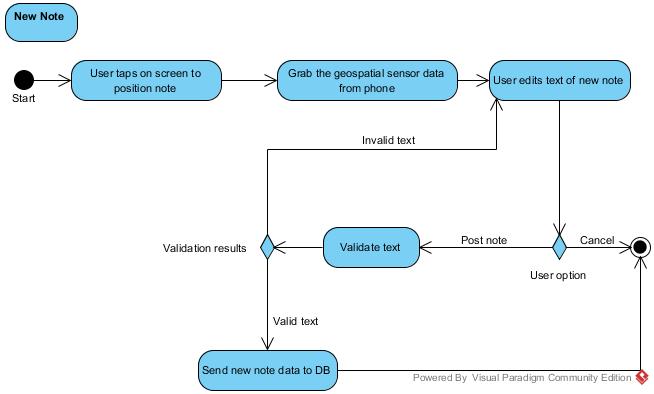
Invisino will confer with the backend database to see if the user has already logged in before and asked to be remembered. If the person has, the database will get the mac address and auto-login the user for them. If they have not checked the Remember Me flag, the person will have to input their information correctly to login. If a person doesn’t have an account, they will have to sign up using the respective button, which opens the register activity. Otherwise, if a person logs in successfully, Invisino goes to the Open Camera Screen activity.



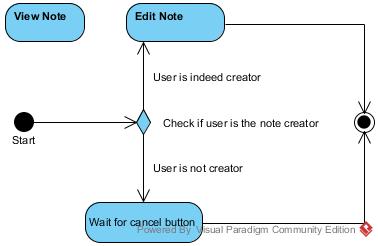
In the case that the user must register, they are prompted to enter their email and password, both of which must be validated, and also confirmed in the case of the password. The user will be notified if either of them are not valid and will have to change them if that is the case. Otherwise, the information they entered is valid. In that case, once the user clicks the register button, the information is sent to the backend database, and the user is now registered. Invisino now goes back to the login screen and automatically logs them in, redirecting again now to the Open Camera Screen activity.



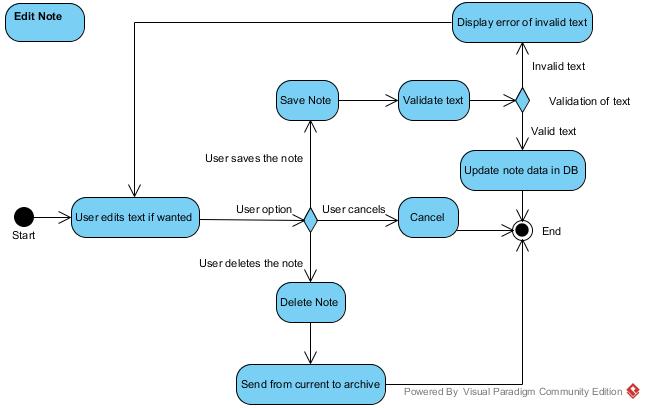
This use case is the main camera screen of the application. Here, the backend database is in routine communication with the device in order to track its location. This way, any and all notes within range of the phone will be picked up and displayed to the user. Moreover, the user has different options available to them. Once again, if the user closes the app, the activity ends and the app stops. If they want to make a new note, it opens the New Note activity.



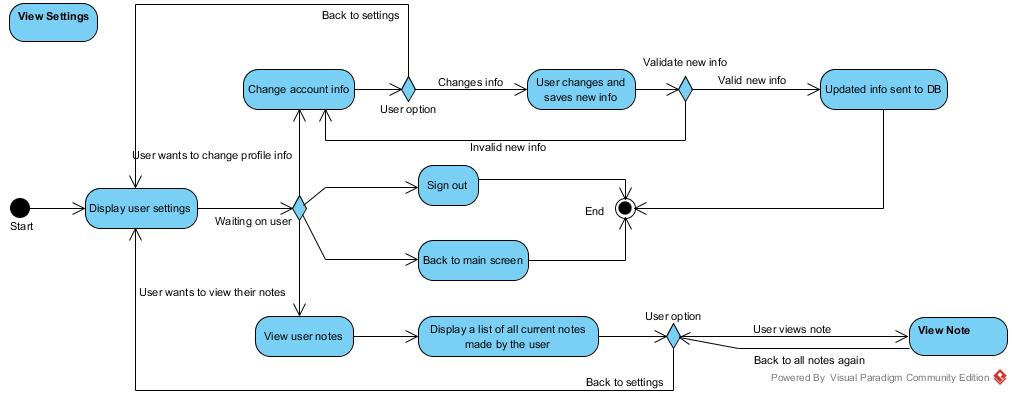
To make a new note, the user must position the note first. Once the note has been put into its wanted location, the Invisino database picks up the phone’s geospatial data to add to the note information. Then, the user fills out text for the note, which must be validated before it can be posted. Once the user finishes making the note, the note data is sent to the database. Then, Invisino returns to the camera screen. There, users also can look at current notes on the screen, when the View Note activity starts.



This use case opens up the note in full screen view for the user to read. From here, the user can cancel the view if they want to return to the main camera screen. However, they can also edit the note if they are the note’s creator, opening the Edit Note activity.

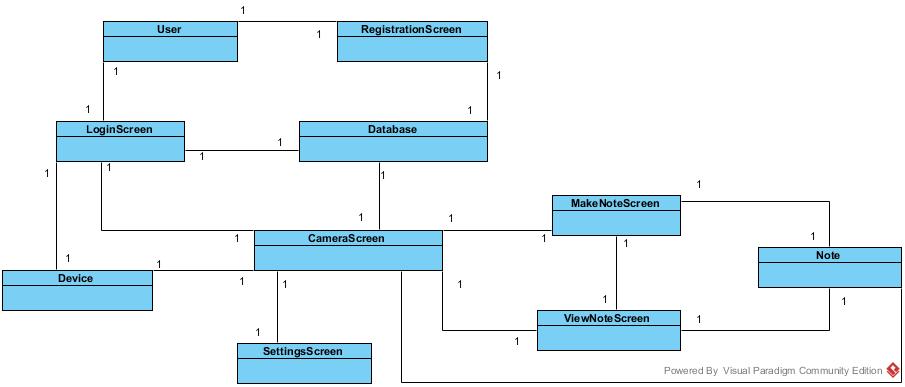


Here, the note creator can edit the text, delete the note, or decide to cancel editing. If they edit the text, it must be validated before the note can be saved. Otherwise, the user will be notified that they must provide text that fits the requirements. Only once valid text has been entered will the note data in the database will be updated. If they want to delete the note at any time, the note will be sent from current notes to the archive and will be no longer visible or even accessible. The user then goes back to the camera screen. On that main screen, if the user wants to see their profile, they open the View Settings activity.



On the settings/profile page, the user has several options. If they want to edit their account info (email and/or password), whatever they change must be validated before it is sent to the database. If they want to view all of their own notes, a list of their currently existing notes will be displayed. The user can view (and edit) any of these notes as wanted. At any point during these options, they can also hit the back button to get back to the main settings screen. If they want to sign out, they are automatically redirected to the login screen because they are no longer in their account. There is also a back button on settings to return to the main camera screen.

1. **Class Diagrams**
   1. **High Level**



The above diagram is the high level class diagram for Invisino. One important thing to note is that any line with no multiplicities is automatically assumed to be a 1-to-1 relationship. Any other multiplicities are explicitly defined.

* 1. **Detailed**

Detailed class diagram to be created in the design phase.

1. **Sequence Diagrams**

No sequence diagrams for the requirements phase, to be continued in the design phase.