Map View will initialize when the app opens or when the user chooses to view the map. Upon initialization, a geotag of the current location is taken. The backend is then sent a Rest post with the geotag data. The backend polls the database to lookup nearby beacons. The database returns the nearby beacons. The backend serves the beacon data to Map View and the map is displayed.

From Camera View, the user will take a picture, verify the picture and optionally add a text description of the picture. A geotag of the current location is also taken. Camera View will take the new picture, text, and geotag and format it into a Rest post and send the data to the backend. The backend will poll the database to create and save the new beacon. Once the database has saved the beacon, a confirmation is returned. The backend then returns a success message to the user. The new beacon is also opened in thread view for the user.

Thread View is opened after posting a new beacon or when a beacon is otherwise selected. A Rest post is sent to the backend to notify the backend which beacon needs to be found via Post ID. The backend polls the database to lookup the beacon. Once found the database returns the beacon data to the backend. The backend then polls the database for the corresponding comments. The comments are returned to the backend. The backend bundles the beacon and comments together and serves them to Thread View. Thread View then displays to the user.

After a post has been selected by a user, they then choose to heart the post. The request is sent to the backend via a Rest post. The backend polls the database to determine if the post is able to be hearted by that user. If the request is valid the database updates the post and returns success to the backend. The backend confirms the action of hearting to the selected post. The heart is then shown to be filled to the user. If the request to heart is invalid, the database returns an error. The backend serves the post with an error, and the user is notified that the action was not valid. These steps are the same for hearting and unhearting a post, but there will be separate URIs.

From Thread View, the user selects to post a comment. Once the user has selected post comment, they enter the text that will be their comment. The new comment is sent to the backend as well as the user ID via Rest post. The backend sends the data to the database to be saved. Upon saving of the comment, the database returns a confirmation to the backend. The backend notifies Thread View that the comment was successfully added. Thread View displays a success message to the user.

Profile View is opened when a user selects to see a user’s profile. A Rest post is sent to the backend with the user ID of the user being viewed. The backend polls the database to lookup the user. The database returns the user data. The backend then polls the database for the user’s posts. The database returns the user’s posts. The backend then serves Profile view with the user and post data. Profile view is displayed to the user.

From Profile view, the user selects follow. The request to follow is sent to the backend. The backend polls the database to verify that the request is valid. If the request is valid the database updates the users and returns. The backend then confirms the follow to Profile view. Profile view displays a success message to the user. If the request is not valid, the database returns an error to the backend. The backend serves an invalid response to Profile view. Profile view displays an error message to the user.

After a post has been selected by a user, they then choose to flag the post if it is offensive. The request is sent to the backend via a Rest post. The backend polls the database to determine if the post is able to be flagged by that user. If the request is valid the database updates the post and returns success to the backend. The backend confirms the action of flagging to the selected post. A success message is then displayed to the user. If the request to flag is invalid, the database returns an error. The backend serves the post with an error, and the user is notified that the action was not valid.

When the app is opened for the first time by a user, the user will be prompted to enter a username and to verify with a third party account. The third party API will then be sent an authentication request. Upon return of authentication of the account, the backend will be sent the new user information. The backend will poll the database to add the new user. The database will return success once the user is saved. The backend then serves the app with a success, and Map view is launched for the user.