Full Design

Due Mon, 15 Nov 2021 at 11:59 pm ET

Commit Document: `project-design`

Submission Form (Also submit commit hash and relevant links on canvas) - One Submission Per Team

After completing the conceptual sketch, you'll work on the detailed design, which you will record in a single integrated design document. This document should include:

- Overview
- Conceptual Design
- Wireframes
- Design Commentary
- Ethics Protocol Analysis

Throughout the design document, you should try hard to be as clear and succinct as possible. You will be penalized for including irrelevant points, for repeating yourself, for saying things in text that are already said in diagrams, and for lack of structure. Feel free to use a tabular form when appropriate: it can be a very helpful way to organize your thoughts.

Every team member should participate in the work behind each section of the design document. The writeup, however, will be divided up, with each section led by (and explicitly attributed to) a single author.

Overview - Karen

This is an outline of what you plan to build. It should include the following:

- Brief description of system to be built
- Key purposes (What problems does it solve? Why should it exist?)
  The purposes could be clearly delineated, each with a name and summary in a short sentence, and then explained more fully.
- Deficiencies of existing solutions (if they exist, and if not, why you suspect not)

Description

We are building a system that allows bikers in Boston to report bike lane blockages by simply dropping a pin on a map and a brief description of why what's causing the blockage. We will record the time and location of the blockage automatically as well as who made the blockage report so that other people can't simply delete a blockage report you posted.

Users can also only resolve/update the status of the blockages they reported themselves but they can comment on any blockage reports of updated statuses or post new blockage reports in the same location with the status set to resolved or unblocked to 'update' the status of a blockage previously reported by another user.

Key Purposes

Reporting Blockage Status

Our app allows users to report paths that are blocked or unsafe, where unsafe means that although the bike lane is not blocked, it is not advised to bike through the lane, perhaps due to snow or slippery ice, etc. We also allow users to mark bike lanes as no longer blocked to update the status of blocked roads. If there's already a blockage report in the bike lane you're currently and it's no longer blocked, you can report a new blockage report with the status set to unblocked in that same pin location, which we will then use our algorithm will determine whether the new resolved report is legit or not, such as voting system and karma. This is important since although we want to notify users when there's a blockage, we also want to let our users know when a road is no longer blocked.

Commenting on the status of blockages

We also allow users to comment on blockage report posts similar to how a thread works in reddit. Users can bond over their frustration over blockages, organize a petition for the city to fix a bike lane, or discuss useful information such as the reason for the blockage, or approximately when the blockage will end. For example, a user could comment "construction worker said that the work should be done in approximately 2 weeks", which is an important update that the original reporter of the blockage can't clarify unless they personally visit the bike lane again and realize that it's no longer blocked.

Subscribe to blockage notifications of specific regions

In addition to just reporting blockages, our app also lets users subscribe to specific regions determined by a (x, y) coordinate location on the map and an radius, as well as a schedule consisted of intervals of time in a similar style as Google Calendar, such as Monday - Friday 9 - 5pm, weekly. This subscription will then give the user push notifications for when there's a new blockage report in that region during those specified times.

Deficiencies of Existing Solutions

Bike Lane Uprising

Bike Lane Uprising is a mobile app that allows users to report bike lane blockages from their phones by submitting information and photos when they encounter a bike lane that is blocked by a car, debris, or other hazards. However, a deficiency with this app is that there could be multiple reports in the same area or bike lane, which can get confusing. It would be nice to have a way to aggregate the reports in the same area somehow in case the blockage or no blockage reports conflict with each other to tell which blockages are real and which ones are made up/trolling.

However, there's no way to resolve a blockage and their UI doesn't give enough information for users to make effective decisions about their community. For example, they don't show the times for when a blockage was reported nor the user and their karma who reported the blockage. Also, there is no voting system for determining if a blockage report was legit so users can easily troll with the system and make a bunch of fake, unverified reports. One good thing about the app UI is that users can submit an image as well as the license plate of the car that's blocking the road, if any, which is helpful information for users to determine if a blockage report was actually legit. Also, the blockage hover card only shows the blockage report number, what kind of blockage it is, and a picture, nothing else. Our app will show the level of karma of the user who made the post and a star next to it if our algorithm performed on the upvoting poll as well as the karma of the poster determined that the post is trustworthy enough. On this app, there's no way to know if the blockage was trolling/reliable enough since it doesn't show anything on the UI determining the level of trustworthiness such as karma or a voting system algorithm.

Conceptual Design - Everyone

Elaborate the concepts in your conceptual sketch into a full conceptual design, giving for each of the key concepts:

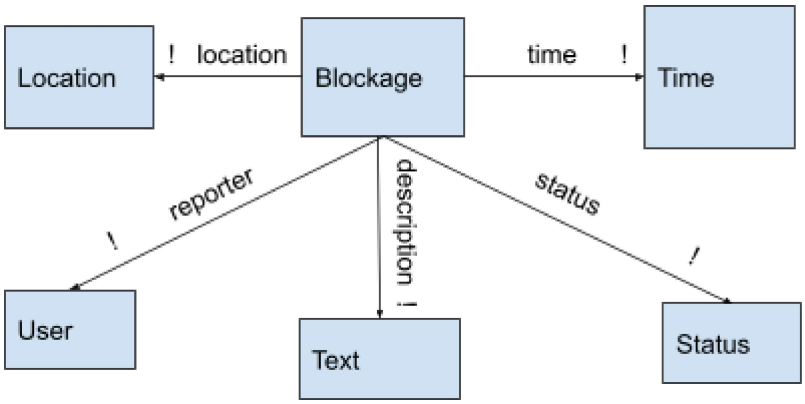
- The name of the concept
- The concept's purpose
- The state, expressed as an abstract data model
- The behavior, expressed as the actions and their effects on the data model
- The operational principle: one or more archetypal scenarios

Your conceptual design should include comments to explain any unusual or potentially confusing elements.

Concept: Blockage - Cindy

Purpose: To track unsafe or unusable bike routes

State:



Actions:

create (l: Location, d: Text, t: Time, u: User, s: Status): Blockage // creates a new Blockage instance

result := fresh Blockage // created new blockage to be in Blockage

result.location := l

result.status := s

result.time := t

result.reporter := u

result.description := d

return result

// users can only edit their own blockages

edit (b: Blockage, d: Text, t: Time, u: User, s: Status) // updates relations of Blockage instance

If b.reporter = u:

b.description := d

b.time := t

b.reporter := u

b.status := s

delete (b: Blockage, u: User) // deletes an existing Blockage instance

when b.reporter == u:

remove b from Blockage

Operational Principles:

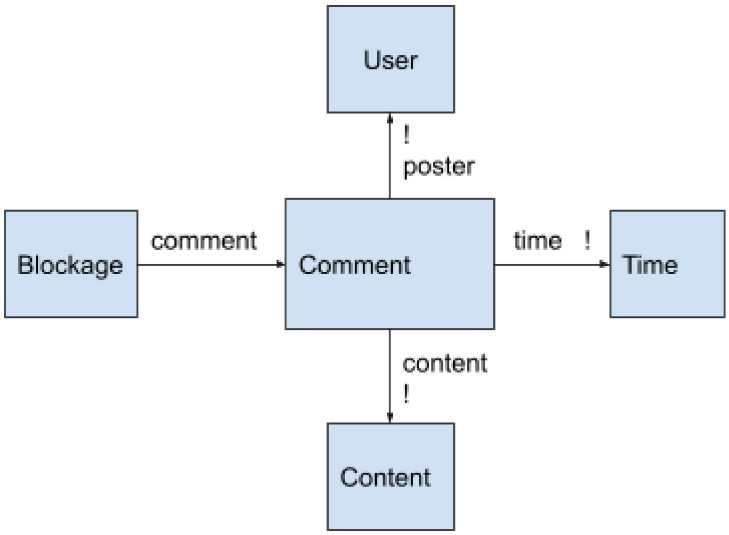
create(l, d, t, u, s, b); edit(b, d', t', u', s') => u = u' // users can only edit their own Blockages

create(l, d, t, u, s, b); delete(b, u') => u = u' // users can only delete their own Blockages

Concept: Comment - Hizami

Purpose: For users to discuss blockages

State:



Actions:

create (b: Blockage, u: User, c: Content t: Time): Comment // creates a new Comment instance

result := fresh Comment // created new comment to be in Comment

result.time := t

result.poster := u

result.content := c

b.comment := result

return result

edit (k: Comment, c: Content, t: Time) // updates relations of Comment instance

k.content := c

k.time := t

delete (k: Comment) // deletes an existing Comment instance

remove k from Comment if k in Comment

Operational Principles:

create(b, u, c, t, k); edit(k, c, t) => u = u' // users can only edit their own Comments

create(b, u, c, t, k); delete(k, u') => u = u' // users can only delete their own Comments

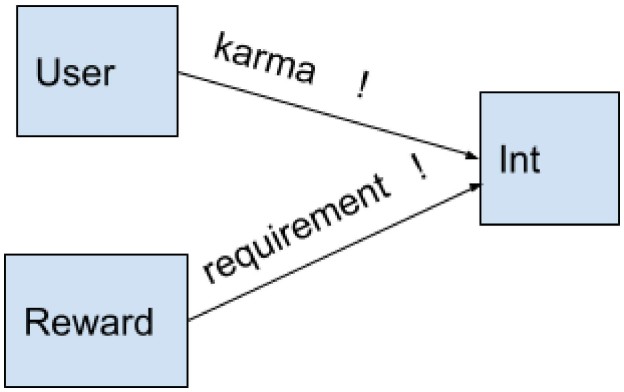
after create(b, u, c, t, k) and delete(b) => not k in Comment

// after you delete a blockage, all of its comments should disappear

Concept: Karma - Karen

Purpose: To privilege good users

State:



karma: User -> one Int
requirement: Reward -> one Int

Actions:

modifyKarma (u: User, n: Int): Int // modifies a user's Karma value
    u.karma += n
setRewardReq (r: Reward, n: Int)
    r.requirement = n
getReward (u: User, r: Reward): Bool // returns true if the user can receive the reward
    return u.karma > r.requirement

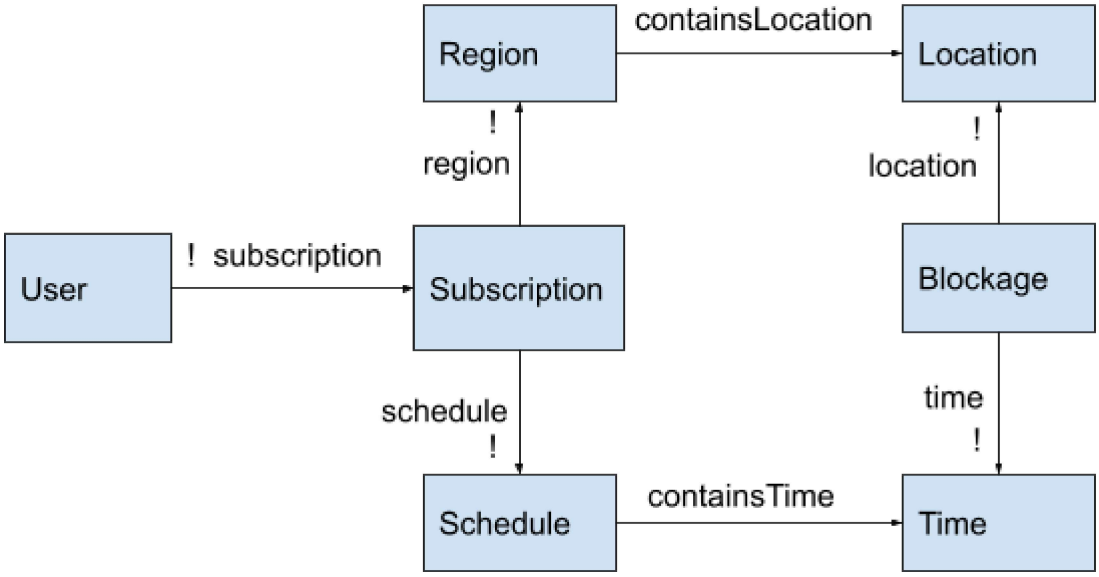
Operational Principles:

If u.karma = 0 (u in User):
    setRewardReq(r, n); modifyKarma(u, n'); getReward(u, r) => n' > n
    // if a user receives a reward, they must have gained more karma than its requirement

Concept: Subscribe - Victor

Purpose: send user notifications for where there's a blockage in their subscribed regions

State:



Actions:

addSubscription (u:User, r:Region, sc:Schedule): Subscription

// user u subscribes to blockages in region r during times given by schedule sc

result := fresh Subscription // created new member of Subscription

result.region := r

result.schedule := sc

result.~subscription := u

return result

removeSubscription (s:Subscription): // remove a subscription

remove s from Subscription if s in Subscription

notify (b:Blockage): set User // determine the set of users to

subs = (b.location.~containsLocation.~region) & (b.time.~containsTime.~schedule)

return subs.~subscription

Operational Principle:

// if the user subscribes to a region r during schedule sc, new blockages will notify the user if its location is in the subscribed region, and its time intersects with the subscribed timeframe.

addSubscription (u, r, sc) | s =>

u in b.notify iff (b.time in sc.containsTime) and (b.location in r.containsLocation)

// if some user adds a subscription and then removes it, this doesn't change whether they're notified of a blockage.

notify (b) **|** us; addSubscription (u, r, sc) **|** s; removeSubscription (s) =>

    notify (b) = us

Sketches - Cindy

You're asked to describe the key concepts of your planned user interface by giving a collection of sketches. These should be organized around your key concepts similar to what you did for A4. Each sketch should depict a screen of your application that shows clearly what elements there are. It should be clear how transitions occur from one sketch to another, and what each element is for. Comments or explanations can be provided as separate notes (making their connection to the elements clear).

You can prepare this sketch by hand, in a drawing tool, or in a specialized wireframing app.

Design Commentary - Hizami

As you work on your design, you will have considered many tricky questions, and will have made decisions between alternatives, sometimes making a tradeoff choosing one imperfect solution over another. This section of your design document brings together all the insights you have gleaned about your design -- why did you settle on the design that you did? It can be presented as a collection of footnotes on the rest of the design document, or as its own free standing text. But however it is expressed, it should discuss substantive and interesting questions, and will be a significant influence on the grade you receive for your design. The prompts you were given in the individual assignments should be helpful to you in determining what kinds of issues to address here.

As always, we encourage you to give a title to each design issue that identifies the problem, and names for each option considered. This is very challenging, but it's a great way to help you focus and ensure that your commentary is well organized and substantive.

Design Issue: Mitigating False Reports

Our app strongly relies on accurate user reports. However, not all users might have the best intentions of our app in mind, and might decide to submit large quantities of false reports, causing inconveniences to our users who believe certain unblocked routes are blocked, and certain blocked routes are unblocked due to these false reports. Our design should try to prevent or at least mitigate the negative effects of these false reports.

Option 1: Users Downvote False Reports

Users have the ability to upvote accurate reports and downvote inaccurate reports. Thus, users know that they can trust reports that have high amounts of upvotes and know to ignore reports that have high amounts of downvotes. The biggest limitation of this option is that it doesn't carry trust over from one report to another. A user who has made a large amount of trustworthy reports can be expected to make more trustworthy reports, and a user who has made a large number of false reports can be expected to make more false reports.

Option 2: Emphasize Reports from High Karma Users

Users still have the ability to upvote accurate reports and downvote inaccurate reports. We introduce a karma system, where a user's karma increases if they receive upvotes, and decreases if they receive downvotes. Thus, users who make more accurate reports will have higher karma and users who make false reports will have lower karma. Then, users know that they can trust reports from high karma users and ignore reports from low karma users. Then, this mitigates a user's ability to disrupt our app with high volumes of false reports from the same account. However, this option still has the limitation that it doesn't actually prevent users from spamming false reports - users are just less likely to trust their accuracy. Furthermore, a user can still spam false reports from multiple accounts.

Option 3: Limit Reports from Low Karma Users

Continuing with the idea of karma discussed in option 2, we will also limit reports from low karma users. For example, an account with near-zero karma may only be able to report up to 3 blockages in a 24-hour period, and an account with highly negative karma might be prevented from reporting blockages altogether. As users gain more karma by making more accurate reports, they gain the ability to report more blockages. Then, users are highly restricted in their ability to spam false reports. One limitation of this choice is that it might be harder for new users to gain karma in order to be able to report more blockages, as reports from users with higher karma will be emphasized over their own reports.

We decided to go with option 3, as it most effectively mitigates the ability of bad actors to spam negative reports which disrupt the functionality of our app. It does limit the ability of new users to gain more karma, but this seems to be an inherent drawback of promoting reports that are more likely to be trustworthy (i.e. from users with higher karma) that cannot fully be eliminated without compromising the integrity of our app.

Design Issue: Locations with High Density of Reports

Due to the fact that our blockages are user-reported, it is very likely that a report that actually has a blockage will receive a large amount of reports. It is possible for this to lead to a very cluttered interface if each report is represented with a separate pin on the map, making it more difficult for users to identify whether certain locations have blockages or not.

Option 1: Display Reports Separately on Map

One option is to display each report separately on the map. A user can zoom in to locations of interest to see the reports more closely. This option has the benefit of users being able to see large amounts of similar reports in the same area, reinforcing the idea that these reports can be trusted. However, when zoomed out, the map will look cluttered and it might be harder for the user to identify locations of interest.

Option 2: Encourage Responding to Existing Reports

Users have the ability to upvote and write comments on existing reports, rather than creating entirely new reports. If a user sees that a report has already been created for the location, they upvote or comment on the existing report rather than creating an entirely new one. Encouraging users to do so, such as by displaying a message when they are about to create a duplicate report, will reduce the amount of reports in a single location. However, our karma system incentivizes users to create new reports in order to receive more positive karma, so it is difficult to rely on users following this policy.

Option 3: Aggregate Close Reports

We can detect when reports are created close enough to each other (i.e. within a 10 foot radius) and aggregate them to the same pin on the map (which could display the number of reports represented by that pin). This will greatly reduce the number of pins on the map, and make it much easier to determine whether or not there is actually a blockage in a given area, especially if there is a mix of positive and negative reports.

We chose option 3 because having aggregated close reports makes the interface look more clean by reducing the amount of pins on the map, allowing users to more easily and quickly identify whether the location they are interested in has a blockage. This choice also doesn't rely on the users for proper functionality, i.e. trusting the users to not make duplicate reports, which allows us to guarantee greater credibility in our report system.

Design Issue: Resolving Blockages

Most blockages will only be temporary. So, after a blockage is reported, we need a way to tell when the blockage is resolved. Ideally, our system will be robust to false reports that blockages have been cleared (misleading users into thinking a blocked route is no longer blocked). However, it will allow for blockages to be correctly detected as “resolved” as quickly as possible, so that users know they can utilize these routes immediately.

Option 1: Lack of Blockage Reports Indicates Lack of Blockage

If there are no more blockages, we can expect that blockages will no longer be reported in this area. After a certain amount of time, blockage reports will expire if blockages do not continue to be reported. A critical flaw of this choice is that if users know that a route is blocked, they will usually not willingly take that route. So we expect the amount of blockage reports for a blockage to decrease over time even if the blockage still exists. Thus, the accuracy of this method of resolving blockages is questionable.

Option 2: Users Vote To Resolve Existing Blockages

On an existing blockage report, we give users the option to mark it as resolved. After a certain amount of votes, the status of the blockage is updated to “resolved”. The biggest limitation of this option is that it is unclear how to deal with multiple blockage reports for the same location - do they all have to be resolved individually? Do we automatically detect that they are close together so that when one is resolved, then all other nearby blockages are resolved?

Option 3: Users Create New “Unblocked” Blockage Instance

Instead of marking existing blockage reports as resolved, users report that there is a lack of blockage in a given location. Then, when the weight of unblocked reports sufficiently exceeds the weight of blocked reports in a location, we can conclude that the blockage has been resolved. This option has the benefit that since we want users to be able to report lack of blockage as well as the existence of blockage, they don’t have to first check whether a blockage already exists in an area to be able to submit their report. However, it has the limitation that it might be difficult to tell whether two blockage reports correspond to the same location.

We chose option 3 as we believe it to be the most reliable. We cannot rely on the idea that lack of blockage reports indicates a blockage is resolved, or on users to not submit duplicate reports. By automatically incorporating new reports that the blockage is unblocked, we mitigate the potential for reduced functionality due to uncooperative users or user error. We also promote the idea that users should submit the fact that no blockage exists in an area even if there was never a blockage to begin with, allowing users to be more certain that the route is safe for biking.

Design Issue: Karma System Unfriendly to New Users

Since loyal trustworthy users might stop biking or reporting blockages in an area after a certain period of time, it is important that our app accommodates a steady stream of new users reporting blockages. However, due to our karma system, it is easy for a user to get discouraged from posting if it is too difficult to gain enough karma for them to be a well-respected user, especially if they are intimidated by other users who have larger amounts of karma. Furthermore, if a new user makes a mistake in a report, it could completely tank their reputation.

Option 1: Displaying Karma Amount

We could display a user's exact karma amount, which gives an extremely accurate measure of how much a user can be trusted. However, this will discourage new users who see that some other users already have thousands of karma and believe that there is no way for them to catch up. Also, users are less likely to respond to reports from users with low amounts of karma, making it even harder for them to gain karma.

Option 2: Displaying Karma Level

Rather than displaying a user's exact karma amount, we display the level of the user which is based on their karma amount, having a maximum level for the most trusted user, and negative levels for untrustable users with negative karma. Then, there is still a reasonably accurate measure of how much a user can be trusted, but the disparity between veteran trusted users and new users is less apparent, as they will only differ by a few levels rather than several thousand karma. One small limitation of this option is that once a user reaches maximum karma level, they might no longer have incentive to post accurate blockage reports frequency. Though, in theory, most users that reach maximum karma will likely be good samaritans anyway.

Option 3: Cap on Negative Karma Per Report

We could put a cap on the negative karma that a user can receive from a single inaccurate blockage report. When a user accidentally makes a mistake in a blockage report, such as incorrect location, and receives a large amount of downvotes, their reputation will not completely tank if they are a new user. The downside of this option is that it makes it more difficult to identify users who are intentionally spamming false reports.

We decided that we would display the karma level of a user, as it preserves the ability for users to judge how trustworthy certain reports are, while mitigating new users being intimidated by other users with large amounts of karma. Furthermore, we decided to put a cap on the negative karma that a user can receive from a single post, so that a single mistake does not tank a new user's reputation. Thus, our app will be more sustainable over a longer period of time, as newer users are more likely to start and continue posting reports of blockages.

Design Issue: Ability to Edit and Delete Reports

As our users are human, they are prone to errors. Thus, we should provide opportunities for users to correct errors, for example if they accidentally create a blockage that they don't mean to, or if they made a typo in the blockage they reported. However, since the trustworthiness of a report is dependent on how many people upvote it, we do not want to risk a user updating a true report to a false report after several people have already upvoted.

Option 1: Users Cannot Edit or Delete Reports

One option is just that users can never edit or delete reports after they are submitted. If a user submits a report with inaccurate information, this will become apparent after the report is downvoted by other users, theoretically mitigating the impact of the misinformation. However, if a user with high karma accidentally makes a typo when submitting a blockage report, it is more likely that a significant number of users will believe the inaccurate information and be negatively impacted, when the reporter would have likely been willing to correct the misinformation if given the chance.

Option 2: Users Have Unrestricted Ability to Edit and Delete Reports

Another option is that users can edit and delete reports anytime after they are submitted. Thus, if a user accidentally submits a report with inaccurate information, they are able to either edit or delete the false report. However, this option has the downside that a user might edit an accurate report's information after several users have already upvoted it, and update it to an inaccurate report or even delete it, compromising the integrity of our report system.

Option 3: Users Have Restricted Ability to Edit and Delete Reports

The final option is that users can only edit and delete reports for a limited duration of time after they are submitted. We might even increase or decrease this duration based on the amount of karma that a user has, allowing more trustworthy users to edit their reports for a longer duration of time after creation. Then, a user has the ability to correct inaccurate information immediately after creating a report, and cannot change information in a report after a large number of people have already upvoted the report.

We chose option 3, as we believe it is the option which will best promote the integrity of reports on our site. By allowing users to edit and delete reports shortly after they are created, we mitigate the amount of false reports created due to user error. By preventing users from editing and deleting reports after a significant amount of time has passed, we mitigate the amount of influence of false reports due to users editing an accurate report after it has already received a large amount of upvotes.

Ethics Protocol Analysis - Victor

You should create an ethics protocol analysis for your project, with discussion of three value-laden design decisions. An explanation of analysis with the ethics protocol and examples may be found in the slides for the lecture here (TBA). You should address every point outlined in the examples from the lecture in a concise and thoughtful way, and you will be asked to update the protocol for the submission of MVP and Finished Product with changes you've made to the protocol after having iterated on your design.

Envision Possible Futures

A good scenario: Safebike Boston launches and quickly gains traction with bikers who are sick of uncertainty in their biking routes, and who want to help each other out. Such a large community forms around Safebike Boston that local governments take notice and pledge to fix neglected, unusable bike lanes, give advance notice of planned construction and events, and rework the layout of streets where cars often drive or park in the bike lane. We begin providing anonymous data to local governments to inform which areas are most in need of attention, and our app provides a central platform for local governments to distribute notices about planned blockages. Other people realize the value of this type of communication between citizens and government, a number of apps are spawned in the same spirit, and even existing companies decide to work with local governments.

A bad scenario: Safebike Boston launches with a fully-featured Karma system, convenient location tracking to drop pins, and the ability to vote on, update, and contest blockages. It becomes very popular with local bikers on launch, and local governments take notice and begin working with Safebike Boston. We let the governments use all the data we collect, but anonymized, and have our data analyst work with them. However, it turns out that they are secretly using this data in other divisions including Crime, where they de-anonymize the data by determining each user's home address from their usage patterns. One day, our data analyst and our community manager notice that our app is actually causing more biking-related fatalities, due to some users trying to farm Karma efficiently while on their bikes. Under pressure from local governments to keep the app up for data collection, this fact is kept secret, and the app remains in use.

Identify Stakeholders

- Our users
- Other bikers in the Boston area
- Drivers
- Pedestrians
- Event organizers
- Restaurants with outdoor seating
- Underserved communities
- People of different ages (Children/Young Adult/Elderly)
- Non-smartphone users

Identify Values At Play

- Outcome Lens: In what ways does what you're making turn out better or worse for your stakeholders?
- Process Lens: How did the process treat stakeholders?
- Structure Lens: How are the outcomes distributed among different stakeholders? What are the differences in how different stakeholders were treated by the process?

Identify Value-Laden Design Choices

1. How do user reports aggregate into blockages?

Possible Choice	Values promoted?	Values demoted?
Any user can update a blockage status unilaterally	**Outcome Lens:** quickest notification about blockages **Process Lens:** each user feels empowered to update a blockage status **Structure Lens:** users are treated equally	**Outcome Lens:** some false reports may impact users
A consensus of a few users are required to update	**Outcome Lens:** users get somewhat quick, reliable notifications about blockages **Structure Lens:** users are treated equally	**Process Lens:** users may feel untrusted or like they don't have a say
A system that depends on Karma	**Outcome Lens:** users are motivated to contribute more to gain Karma; results are more trustworthy and resilient to bad actors	**Structure Lens:** Users with less Karma may feel less valued

2. How should users be incentivized to report blockages?

Possible Choice	Values promoted?	Values demoted?
No incentive, rely on goodwill	**Process Lens:** users are free to use the app without reporting blockages	**Outcome Lens:** disuse of the reporting feature might compromise the efficacy of the app
Internal incentive in terms of reputation or Karma on the app	**Outcome Lens:** users who want reputation will participate more **Process Lens:** users are still not pressured into always reporting/updating	**Outcome Lens:** some users may still never report or update blockages **Structure Lens:** may widen divide between active reporters and lurkers
Design tactics such as notifications to remind users to update status of blockages	**Outcome Lens:** efficient status updates for existing blockages	**Process Lens:** likely annoying and possibly unsafe for users

3. What information about a user should be readily available to others?

Possible Choice	Values promoted?	Values demoted?
All reports anonymous	**Process Lens:** user privacy is fully respected **Structure Lens:** users are treated equally	**Outcome Lens:** a sense of community is difficult to develop, which may reduce efficacy of the app
Basic info, such as username, Karma, how long they've been on the app	**Outcome Lens:** facilitates a sense of community **Process Lens:** no sensitive information about a user is made public	**Outcome Lens:** users still don't know much about others, and may not trust them **Structure Lens:** these create a distinction between new and experienced users
History of reports/comments, their preferred routes	**Outcome Lens:** more ways to develop community; users can decide for themselves who to trust	**Process Lens:** user privacy is compromised

4. How will we deal with users who post malicious blockage reports?

Possible Choice	Values promoted?	Values demoted?
Don't restrict the user; rely on Karma to hide spam reports	**Process Lens:** Every user is given the benefit of the doubt **Structure Lens:** Users are treated equally	**Outcome Lens:** If reports still visible before they get downvoted, then users can still post many bad reports
Automatically ban users based on certain conditions	**Outcome Lens:** Removes users who consistently post misleading blockage reports **Structure Lens:** Users are treated equally by an automatic system	**Process Lens:** May scare users due to the strictness, and fails to deal with edge cases such as getting hacked, or "sibling took my phone"
Restrict users' activity if they meet certain conditions, and allow them to submit an appeal to staff	**Process Lens:** Feels more personal and less fixed, so less likely to scare users **Outcome Lens:** Achieves nearly-identical outcomes to the hard ban method	**Structure Lens:** Staff may introduce bias into the appeals process

Choose & Justify

1. How do user reports aggregate into blockages?

Choice: A system that depends on Karma

- By considering each user's Karma, we ensure that each blockage report is reliable, unlike the solution where any user can create a full blockage report
- This efficiently ensures the reliability of blockage reports, unlike the solution which requires the agreement of a few users, where even very trusted users cannot report a blockage by themselves.
- This incentivizes being a longtime, active user of the app, and active users are the users who make the app function
- If handled insensitively, this may drive new users away from being active on the app, since they feel like they have less of a say, and may be judged for being new

2. How should users be incentivized to report blockages?

Choice: Internal incentive in terms of reputation or Karma on the app

- By having a karma system that rewards active participation and accurate blockage reports, we encourage greater user engagement which improves reliability of our app
- Users can feel more confident in their ability to trust blockage reports from other users if they see that the reporter has high karma and a history of reliable reports
- Users are not pressured to update every blockage status, unlike with the notification solution, which lets them only report/update when it is safe for them.

3. What information about a user should be readily available to others?

Choice: Basic info, such as username, Karma, how long they've been on the app

- Basic information is necessary to create a basic feeling that there are people using the app, and to tell at a glance how trustworthy/experienced a user is
- Users' accounts and associated information aren't linked with their identities by default, and this makes it easier for it to stay that way.
- In particular, this makes malicious behavior such as stalking or doxxing much more difficult to do via SafeBike.
- Not having much information about users may make it harder to engage in a community, but having attached usernames helps, and a user-editable "about me" might mitigate this.

4. How will we deal with users who post malicious blockage reports?

Choice: Restrict users' activity if they meet certain conditions, and allow them to submit an appeal to staff

- With the right conditions, can effectively prevent loopholes that allow malicious users to continue using the app
- Deals with the fact that regardless of how we determine blockage reports are malicious ("report as malicious" feature, or a negative Karma threshold), we can't have full confidence, especially when detection is automated and there may be edge cases.
- If the userbase grows to a point where staffing appeal reviews becomes unmanageable, we could nominate community moderators

MVP Changes - Karen

Our mentor suggested we add another core concept besides karma, such as letting the user bookmark their favorite routes. In response to this request, we implemented the Subscription/Alert concept that allows the user to select certain times during days of the week to get alerts for new blockages from. Although we haven't implemented the notification drop down yet, the user can now add new alerts in the "My Alerts" tab by selecting a center and radius along with the times of the week to receive alerts from. The UI will display all of the blockage reports within the user's subscribed areas for the user to see. We also allow the user to give a title for each of their newly created subscriptions to help them remember what the specific alert was for, such as "work" where the schedule can be 9am - 5pm every weekday and "yoga classes", where the schedule can be 5 - 7pm Monday, Wednesday, and Friday. The work alert will eventually be able to alert the user when there are new blockages or status changes for blockage reports in that indicated area during those times of the week so the user can plan to change their bike route accordingly to save time and energy in the morning when they're rushing to get to work.

For the blockage reports, we've also implemented the comment feature, update status feature, as well as the view history option. The comment feature is rather self explanatory, where users can comment on posts including their own when logged in, as well as see the comments of other people even when not logged in. Additionally, we also have an update status button now for each of the blockages to allow other users, including the original reporter of the blockage post themselves to update thes status and description of each of the blockages, such as marking an original blockage as no longer blocked, for example, when the construction on the road cleared up and the original user wasn't there physically to witness the change in status themselves.

A feature that comes with the update status is the view history button, also at the bottom of the blockage sidebar, where the user can see a list of all of the previous updates by themselves or other users, sorted in chronological order from the most to least recent. Each of these update status histories show the user who made the update, the report's updated status by that user, the time they made the update, as well as an optional new description, if any, such as "construction finished", if they changed the status from "BLOCKED" to "UNBLOCKED", for example.

Finally, we also implemented a simple upvote/downvote for our karma concept, which we titled as activity points, which can be accessed from the profile drop down popup in the top right corner when you click on the profile icon in the nav bar. Currently, users can upvote, downvote, or do neither for each of the posts on the map, including those of their own. Upvoting a blockage report means that the user confirms that the report is accurate and downvoting implies that the blockage report is not accurate, such as a BLOCKED report when the bike lane was confirmed by the user to not be blocked. Next to the upvote and downvote buttons in the bottom right corner of each of the posts is the total vote count, which is a number representing the net vote

count of the post, with a + 1 for each upvote the post received and - 1 for each of the downvotes the post received. The higher the total vote count the better the karma of the reporter gets. If the total vote count is negative for the post, then the reporter will have a decrease in their karma in the amount indicated by the total votes just from that post alone.

For each of the upvotes the user gets for their own blockage report posts, they get an activity point for it and each downvote corresponds to minus one activity point for the user who made the post. The more activity points the user has, the more karma they have and therefore the higher their ‘level’, which is also displayed in the top drop down menu right above the activity points label.

All new users start at level 1 with 0 activity points and once they get to 3 activity points (which can either be due to other people’s net vote on their posts, their own votes on their posts, or a blend of both), the user will have earned enough points to be at level 2 karma, which will eventually give them more power such as making more blockage reports or update statuses in the finished product.

Heuristic Evaluation - Cindy

1. Visibility of System Status

The design should always keep users informed about what is going on, through appropriate feedback within a reasonable amount of time.

The pages have helpful headers that display what the user is currently viewing. When a user is reporting a blockage, the form displayed is labeled with "Report Blockage". The user should never be confused about what they are currently looking at on the page.

2. Match between System and the Real World

The design should speak the users' language. Use words, phrases, and concepts familiar to the user, rather than internal jargon. Follow real-world conventions, making information appear in a natural and logical order.

Our UI does not use complicated jargon, using simple and standard phrases to describe functionality such as edit and delete. The information that users are most likely to need are displayed right on the home page.
We use a navbar across the top, which is a very normal design choice in context of existing popular apps. All of our icons in the navbar are accompanied by their corresponding titles given enough space, so they are not confusing. My reports are shown in a logical chronological order, and my alerts are shown in order of creation. The pins are also color coded in a conventional way, where green means unblocked, red means blocked, and yellow is unsafe (but usable with caution).

3. User Control and Freedom

Users often perform actions by mistake. They need a clearly marked "emergency exit" to leave the unwanted action without having to go through an extended process.

Blockages: Users have the ability to edit and delete blockages they have created.
Subscriptions: Users can update and delete subscriptions they have created. If a user accidentally activates the feature to create or update a subscription, they can click a cancel button to deactivate it.
Accounts: Users have the ability to change their username, change their password, or delete their account. If a user accidentally clicks on the button to delete their account, a warning pops up that gives them a chance to cancel.

4. Consistency and Standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform and industry conventions.

Our UI uses standard platform conventions, such as a pencil icon indicating an edit feature, and a trash icon indicating a delete feature. We use standard vocabulary for pages such as Home, Account, Settings, Login, Logout.

5. Error Prevention

Good error messages are important, but the best designs carefully prevent problems from occurring in the first place. Either eliminate error-prone conditions, or check for them and present users with a confirmation option before they commit to the action.

Submission forms prevent the user from submitting if the input is not properly formatted. For example, if they leave the username or password fields blank on the signup/login form, then the button to submit the form is greyed out until the fields are no longer blank

6. Recognition rather than Recall

Minimize the user's memory load by making elements, actions, and options visible. The user should not have to remember information from one part of the interface to another. Information required to use the design (e.g. field labels or menu items) should be visible or easily retrievable when needed.

Most actions are immediately visible on the surface of the UI. No actions are hidden behind complex maneuvers or sequences of keypresses, and most actions don't even require the user to navigate a menu.

7. Flexibility and Efficiency of Use

Shortcuts — hidden from novice users — may speed up the interaction for the expert user such that the design can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Our UI is designed to be easy and convenient, taking users only a few seconds to report blockages and to view blockages. Thus, there is very little about interaction with the website that requires shortcuts that speed up the process.

8. Aesthetic and Minimalist Design

Interfaces should not contain information which is irrelevant or rarely needed. Every extra unit of information in an interface competes with the relevant units of information and diminishes their relative visibility.

Our UI displays the most relevant information about blockages to the users. Irrelevant information like Blockage ID or Reporter ID are not displayed, as these are only useful internally and don't matter to users. Blockage information is hidden by default, so only the type of blockage (indicated by the colors of the pins) and the location (shown on map) are immediately visible. Users can access extra information about any blockage in one click.

9. Help Users Recognize, Diagnose, and Recover from Errors

Error messages should be expressed in plain language (no error codes), precisely indicate the problem, and constructively suggest a solution.

Our UI displays easily-understandable error messages that tell the user exactly how to fix the error, such as "Usernames cannot be more than 15 characters long" (no error codes or complicated jargon).

10. Help and Documentation

It's best if the system doesn't need any additional explanation. However, it may be necessary to provide documentation to help users understand how to complete their tasks.

The system provides helpful instructions that tell the user how to use the functionality of the site. For example, "Double click on the map to create a new blockage." The homepage has a floating block that has simple instructions on it. The subscriptions page also has instructions shown, such as "Click on an existing circle to view details on the subscription."

New Design Issue: Subscription Scheduling - Hizami

When users create a subscription, they need a way to indicate what time slots they would like the notifications to occur. Ideally, the method of scheduling notifications should offer enough flexibility to suit the needs of the user, while still making the process of scheduling notifications as convenient as possible for the user. Flexibility and convenience come at a tradeoff, so the option we choose should have the best balance between the two.

Option 1: Always Receive Notifications

One option is just to not allow users to choose a time frame for their subscriptions. Thus, whenever a blockage occurs in the subscribed region, no matter what time, the user will receive a notification. This option provides the most convenient process of scheduling notifications as they do not need to be scheduled. However, users can receive blockage notifications even at night when they do not plan to be biking, which can be an annoyance to the user who may even choose to disable notifications altogether as a result.

Option 2: Use Calendar to Choose Time Slots

Another option is to use the standard calendar scheduling, where we show all 7 days of the week and 24 hours in grid form, and the user can select the time frames for the subscription. This option provides the most flexibility for the user to select their schedule. However, this method is rather complex and can be inconvenient to select the same time frame on multiple days if the user wants a more regular schedule of notifications, which we expect to be the average use case of our app. Furthermore, it will likely prevent us from being able to conveniently display the scheduling of notifications in the subscriptions list, as displaying the entire calendar would require quite a bit of space.

Option 3: Choose Days and Time Window

The final option is to allow users to select the days of the week and a single time window on those days for the notification. This option provides a balance of flexibility and convenience. We expect our users to have more regular daily schedules (such as working from 9 to 5 on M-F), so this option should provide sufficient flexibility to the average user. However, one limitation of this option is that users who might wish to have more complex notification scheduling cannot conveniently do so. Though, we expect this to be a rare case and users can simulate this by creating multiple separate subscription notifications for the same region.

We chose option 3 because we believe it provides the best balance of flexibility and convenience out of the three options. We believe that it offers enough flexibility for the use case of the average user, while still being a convenient method of scheduling notifications. We might also consider giving users the option to import a Google calendar for scheduling notifications in the rare occasion where option 2 would benefit the user more.

Ethics Protocol Changes Summary - Victor

- We began thinking about a new design issue-- how we should deal with bad behavior or clearly malicious blockage reports and users.
- We rephrased the question, “What information about a user should be public to others?” to “What information about a user should be readily available to others?” instead. This draws a distinction between information that one user can technically find about another user if they look hard enough, and information that sits on a user’s profile page. It also makes the question more about what information we want to *emphasize* about a user, not just what information is available.
- With the above in mind, we decided the best path forward was to only make basic information (e.g. Activity Score/Level) readily available to users. Although all blockages a user has posted is public information, as we display their username on the blockage, it’s never aggregated in an easy-to-browse format.