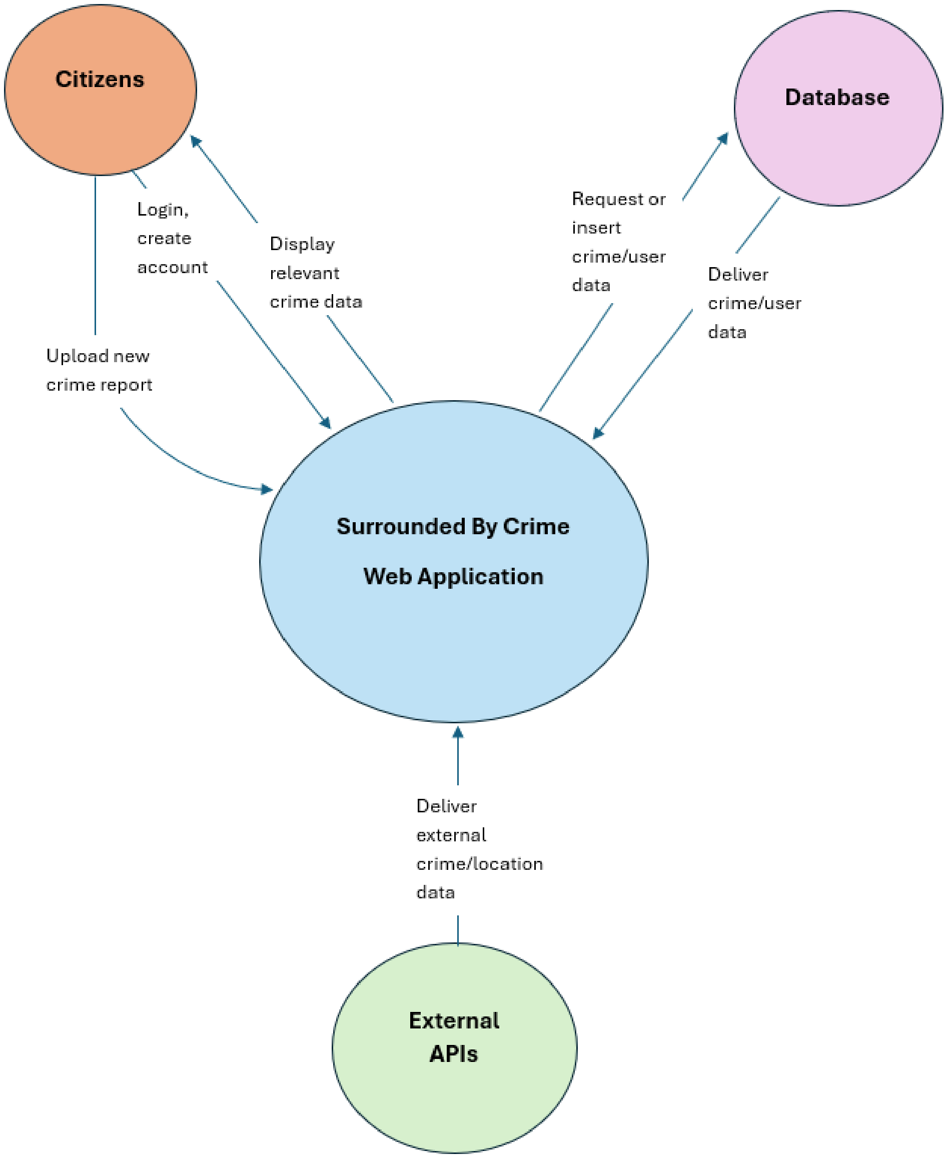
SD^2: Surrounded By Crime

1

Our motivation for Surrounded By Crime came from the lack of public notification about crime. We saw a world where crime is often kept out of the headlines and out of public knowledge. Surrounded By Crime aims to solve this lack of transparency by removing the reliance on police and news outlets for information about crime. With Surrounded By Crime everyday people can report and learn about crime in their area. This allows real people to share the crime really happening to them and for everyday people to keep up to date and aware of the crime happening in their area. This will allow people to live safer lives, more prepared for the dangers actually happening around them.

Context Diagram:


Citizens
Database
Login, create account
Display relevant crime data
Request or insert crime/user data
Deliver crime/user data
Upload new crime report
Surrounded By Crime Web Application
Deliver external crime/location data
External APIs

Main Use Cases:

See A Crime

USE CASE NAME:	See A Crime	**USE CASE TYPE** Business Requirements: □ System Analysis: □
USE CASE ID:	0	
PRIORITY:	1	
SOURCE:	Emil Kozicki	
PRIMARY BUSINESS ACTOR	Server	
PRIMARY SYSTEM ACTOR	Citizen	
OTHER PARTICIPATING ACTORS:	●	
OTHER INTERESTED STAKEHOLDERS:	●	
DESCRIPTION:	This is how a citizen will use the app if they see a crime	
PRE-CONDITION:	A citizen with a Surrounded By Crime account	
TRIGGER:	A citizen has seen a crime	
TYPICAL COURSE	**Actor Action**	**System Response**

OF EVENTS:	**Step 1:** Log into Surrounded By Crime	**Step 2:** Verify Login
	Step 3: click the button that says “new report”	**Step 4:** Shows user the new report screen
	Step 5: Fill in the report form	
	Step 6: submit the report	**Step 7:** upload the report to the database
ALTERNATE COURSES:	● Citizen could already be logged in and thus not need steps 1 and 2	
	● Citizen might be on a different page on the site and need to navigate to the home page	
CONCLUSION:	A new report of the observed crime has been uploaded into the database and now any other users that have the location of the reported crime as part of their distance preference will see the report.	
POST-CONDITION:	**N/A**	
BUSINESS RULES	●	
IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS	●	
ASSUMPTIONS:	● internet connection	
OPEN ISSUES:		

Know About Crime

USE CASE NAME:	Know about Crime	
USE CASE ID:	1	
PRIORITY:	1	
SOURCE:	Scott Souza	
PRIMARY BUSINESS ACTOR	Server	
PRIMARY SYSTEM ACTOR	Citizen	
OTHER PARTICIPATING ACTORS:	●	
OTHER INTERESTED STAKEHOLDERS:	●	
DESCRIPTION:	This is how a citizen will use the app if they want to know about crime in their area	
PRE-CONDITION:	A citizen with a Surrounded By Crime account	
TRIGGER:	A citizen wants to know if any crime has occurred in their area	
TYPICAL COURSE	**Actor Action**	
OF EVENTS:	**Step 1:** Log into Surrounded By Crime	**Step 2:** Server verifies log in

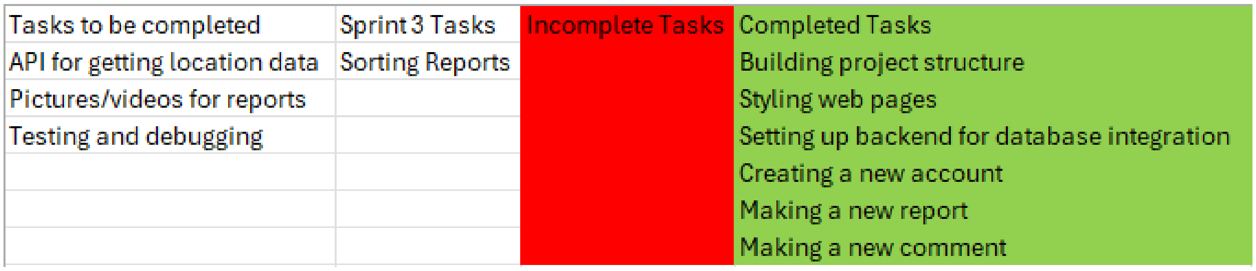
	Step 3: Citizen updates their distance preference	**Step 4:** server updates area preference and the updated reports are sent back to the user
	Step 5: Citizen is now able to view all crimes in their area	
ALTERNATE COURSES:	● Citizen could already be logged in and thus not need steps 1 and 2	
	● Citizen might be on a different page on the site and need to navigate to the home page	
CONCLUSION:	The Citizen is shown a list of all reports of crime in their area.	
POST-CONDITION:	**N/A**	
BUSINESS RULES	●	
IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS	●	
ASSUMPTIONS:	● internet connection	
OPEN ISSUES:		

Question A Crime

USE CASE NAME:	**Questioning A Crime**	
USE CASE ID:	**2**	
PRIORITY:	**1**	
SOURCE:	**Aaron Crandall**	
PRIMARY BUSINESS ACTOR	**Server**	
PRIMARY SYSTEM ACTOR	**Citizen**	
OTHER PARTICIPATING ACTORS:	●	
OTHER INTERESTED STAKEHOLDERS:	●	
DESCRIPTION:	**This is how a citizen will use the app if they wish to ask questions about a crime**	
PRE-CONDITION:	**A citizen with a Surrounded By Crime account and is logged into that account**	
TRIGGER:	**A citizen has seen a crime report they wish to know more about**	
TYPICAL COURSE	**Actor Action**	

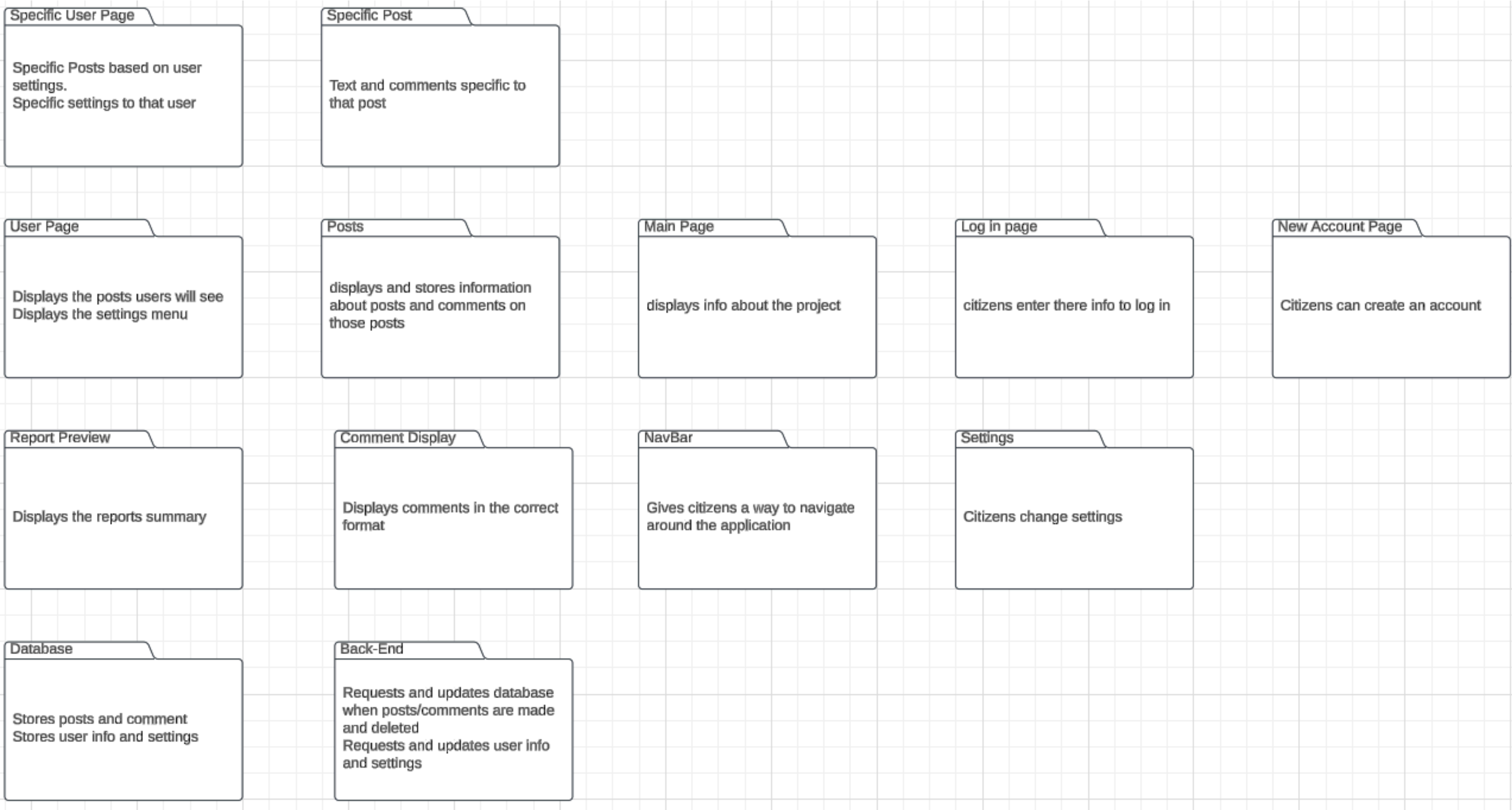
OF EVENTS:	Step 1: Citizen clicks on the report they wish to ask a question about	Step 2: Server takes citizen to the report page
	Step 3: click the button that says “new comment”	
	Step 4: Citizen makes their comment and submits the comment	Step 5: Server uploads the comment.
ALTERNATE COURSES:		
CONCLUSION:	A new comment has been made on the report. The citizen that made the report is able to see the new comment and discuss the report	
POST-CONDITION:	N/A	
BUSINESS RULES	●	
IMPLEMENTATION CONTRAINTS AND SPECIFICATIONS	●	
ASSUMPTIONS:	● internet connection	
OPEN ISSUES:		

Product Backlog



Tasks to be completed	Sprint 3 Tasks	Incomplete Tasks	Completed Tasks
API for getting location data	Sorting Reports		Building project structure
Pictures/videos for reports			Styling web pages
Testing and debugging			Setting up backend for database integration
			Creating a new account
			Making a new report
			Making a new comment

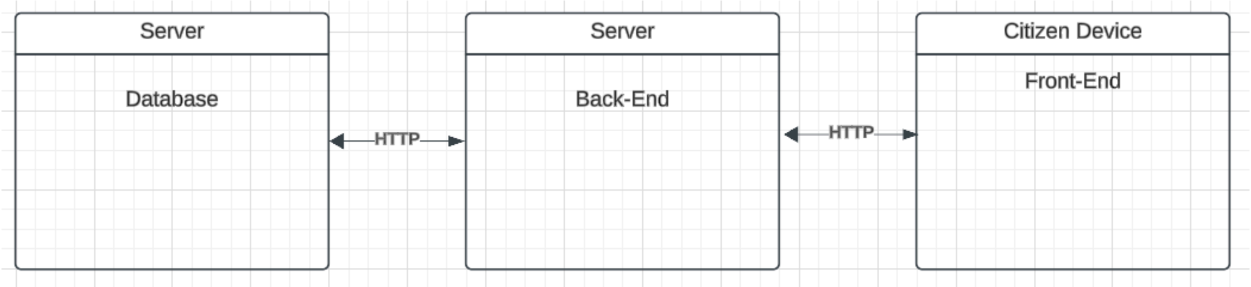
2.1


Specific User Page
Specific Posts based on user settings.
Specific settings to that user
Specific Post
Text and comments specific to that post
User Page
Displays the posts users will see
Displays the settings menu
Posts
displays and stores information about posts and comments on those posts
Main Page
displays info about the project
Log in page
citizens enter there info to log in
New Account Page
Citizens can create an account
Report Preview
Displays the reports summary
Comment Display
Displays comments in the correct format
NavBar
Gives citizens a way to navigate around the application
Settings
Citizens change settings
Database
Stores posts and comment
Stores user info and settings
Back-End
Requests and updates database when posts/comments are made and deleted
Requests and updates user info and settings

Specific User Page is where the citizen will see the posts based on his preferences for date and distance. Specific post is where a citizen will see each post and be able to make comments on them. User page is the generic page for all users without each user's settings. Posts are generic pages for all posts without any of the posts specifics. Main page is the home page and gives information about the application. The log-in page is where citizens go to log in. The new account page is where citizens create new accounts. Report Preview is a reusable component that formats the reports for viewing in the user page. Comment Display is a reusable component that formats the comments for viewing in the posts page. NavBar gives citizens the ability to navigate around the application. Settings is a page where a citizen can edit their user settings from inside the user page. The database stores all of the data on citizens and posts with comments. Back-end handles all of the requests and updating of data.

We use component based architecture for our project. We chose this architecture because we realized that our application would want to look the same for every citizen. The only difference should be the data that is being displayed to each citizen. Thus we needed to build reusable components that would each look the same but display data that was unique to every citizen based on their preferences.

2.2



In our project we have three different devices running. There is a database which will run on a remote server. There is a back-end that will run on a remote server. The last is the client side which is run on the citizen's client. All of the connections are internet based HTTP because that is what we had the most experience with. The back-end and the database could in theory be run on the same server.

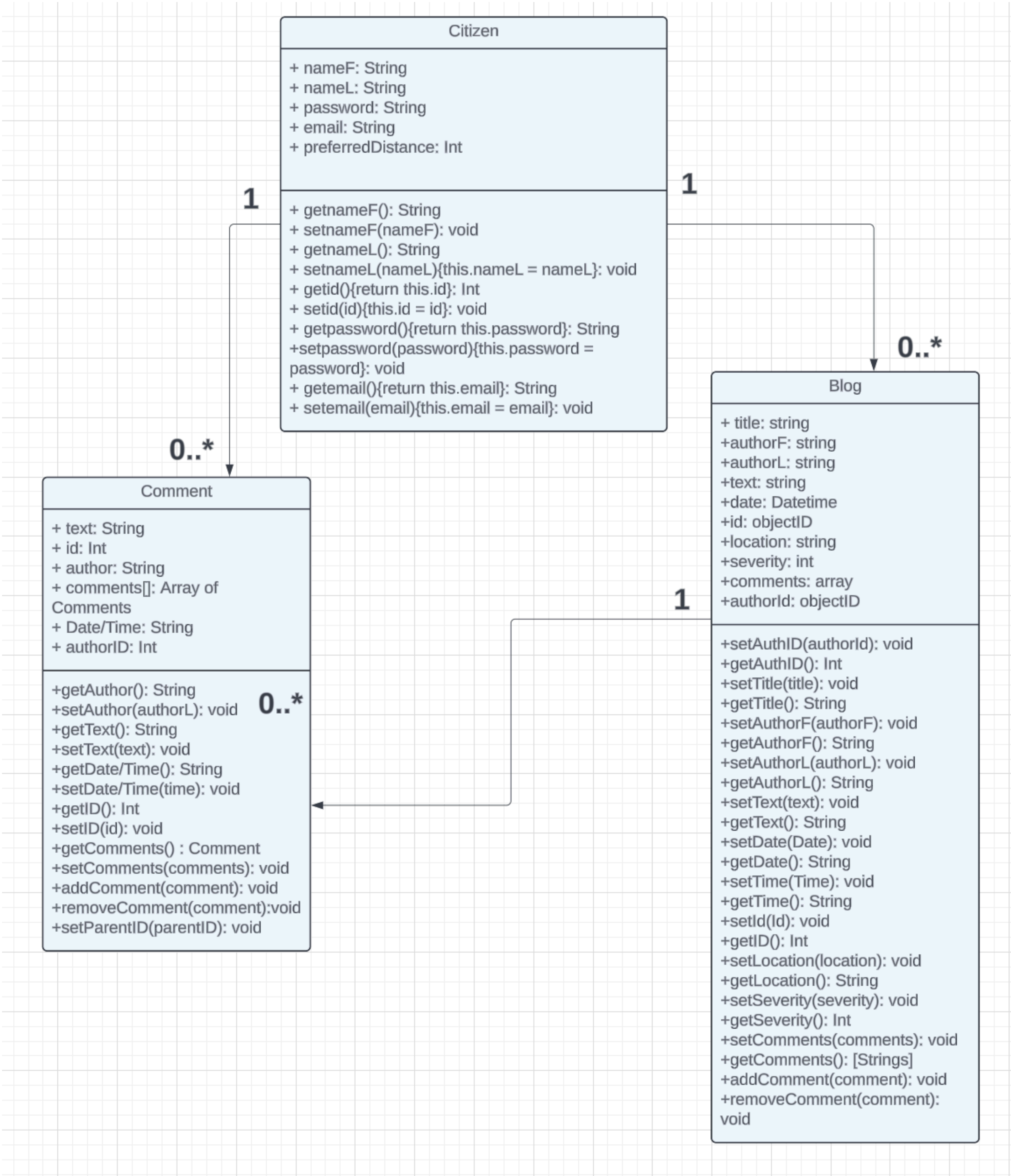
2.3

Our project does use persistent data storage. We chose to continue to use MongoDB as our database of choice, which stores documents instead of storing information within tables. The two data points that require storage in our application are blogs (or reports) and users. The stored fields within blogs are its title, author first and last names, the text description of the crime event that took place, comments, the date and time, location, severity, and its associated object ID. This object ID can be used to uniquely identify the blog later on when users want to comment on the report and it needs to be updated in the database. Our user data point stores the basic information you would expect to be asked to provide when creating an account: first name, last name, email, and password. All users are also given a unique object ID upon registration for further data upkeep.

2.4

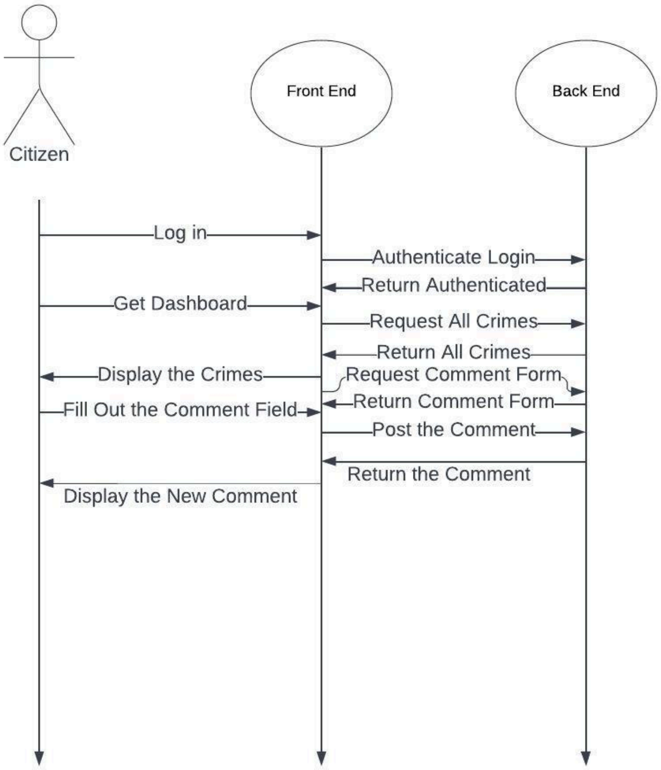
The control flow for our program is mostly event driven with some procedure. When a citizen first goes to our homepage there are a few options for what they might see. If they are logged in they will see a link to their custom page. If they are not logged in they will see links to either log in or create an account. After they either make an account or log in they will get taken to their custom page. From this page they will see the posts that fit their current settings. From here they can change their settings to see different posts, they can click on a post to see more about it or to make a comment, they can make a new post, or they can log out. These four options are in a constant loop until the log out option is selected and then they are taken page to the home page and the loop repeats.

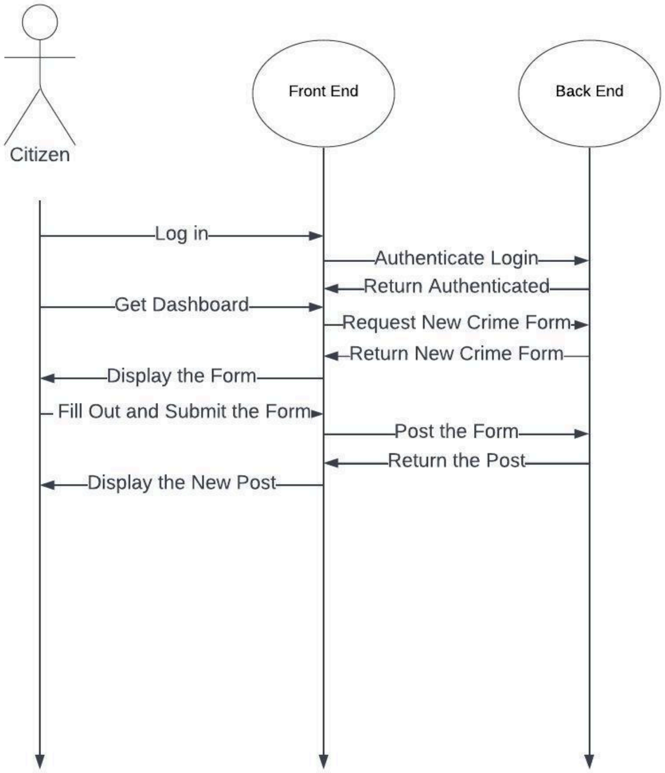
3.1



Pictured above is a UML diagram of classes in our project; it contains three of them: Comment, Blog, and Citizen. The Comment class, in short, creates and maintains citizen comments in each blog. A citizen can comment on each blog and also reply to each comment; each class communicates with the comment class. A blog class is responsible for creating blogs. In order to add a new blog, a citizen has to create an account and be logged in. This class is in close contact with the comment class to make sure all the comments are there. The last is the Citizen class, which is the only class that communicates with all the other classes. It is responsible for creating the users and by being able to communicate with the other two classes, citizens can add blogs and comments.

3.2


Citizen
Front End
Back End
Log in
Authenticate Login
Return Authenticated
Get Dashboard
Request All Crimes
Return All Crimes
Display the Crimes
Request Comment Form
Return Comment Form
Fill Out the Comment Field
Post the Comment
Return the Comment
Display the New Comment


Citizen
Front End
Back End
Log in
Authenticate Login
Return Authenticated
Get Dashboard
Request New Crime Form
Return New Crime Form
Display the Form
Fill Out and Submit the Form
Post the Form
Return the Post
Display the New Post

Two of the most important functionalities of our web app are adding a blog and adding a comment. The first thing the user must do is log into their account. From there, they will be able to add a new blog of their own. What they can also do is comment on already existing blogs. As we can see in both UML sequence diagrams above, there is a lot communication between the frontend, the backend, and the citizen in order to create a new post or add a new comment. Whether it is creating a new blog or adding a comment, the processes are very similar. A user logs in and authentication is performed. Then they are taken to the dashboard where a request is made to show all crimes or display a new crime request. Then, all crimes or a new form is displayed. Once a form is completed it is posted and the blog or comment is displayed to the user.

4.1 Installation directions

Surrounded-By-Crime
Installation Guide

-----In one terminal-----
Run the following commands:

```
cd back-end
npm install
npm start
```

-----In a second terminal-----
Run the following commands:

```
cd surrounded-by-crime
npm install
npm start
```