Police Bot: Enhancing Social Media Governance with Policing Bots

Milestone 1 Presentation

Group Members:

Students:

- Liam Dumbell
- Gabriel Silva
- Cody Manning

Faculty Advisor / Project Client:

- Khaled Slhoub

Computer Science Project Instructor:

- Philip Chan

Overview:

- Comparing Technical Tools
- Discussion on finalized collaboration tools
- Discussion on Requirements, Design, Test and Milestone 1 Progress Evaluation Documents
- Tweepy Demo Video
- Discussion on resolved and new Technical Challenges
- Plans and Adjustments heading into Milestone 2

Comparing Technical Tools:

- Tweepy:

  Allows users to connect to existing Twitter accounts.
  Allows users to Post Tweets on the linked account.
  Allows users to scan individual Twitter account data.

- Flask Application Approach:

  Has similar functionality to Tweepy, but has far more setup involved.
  Decided to be out of the scope of our project due to this approach involving stockpiling access tokens in a database, which gets very expensive.

Collaboration Tools:

- Software Development: Github (most efficient way of storing all relevant code)
- Documents / Presentations: Google Docs, Google Slides and Powerpoint
- Communication Method: Discord
- Task Calendar: Google Calendar

Requirements Document:

Detailed Requirements for the following functionalities of our framework:

- Bot Creation
- Bot Scheduling
- Bot Discovery
- Bot Distinguishing
- Data Storing

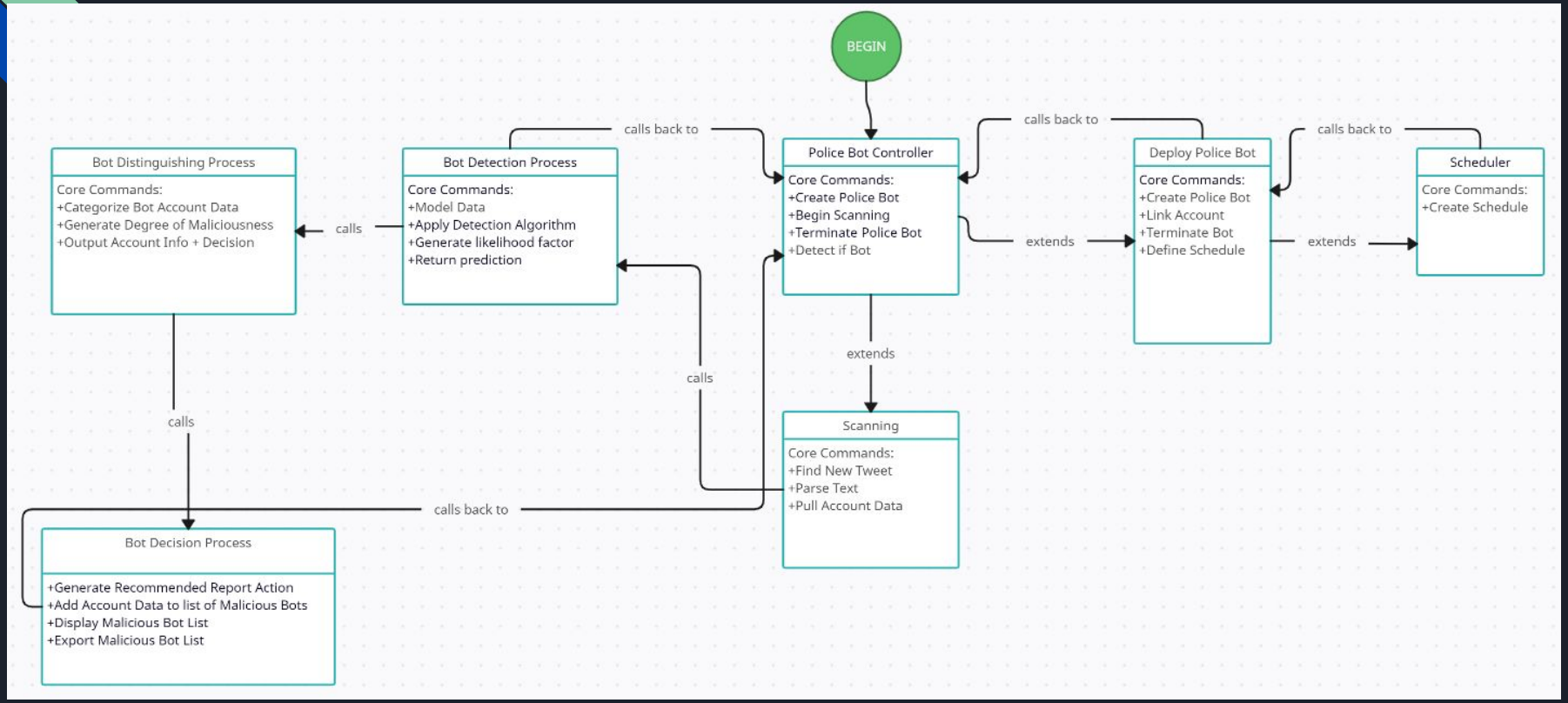
Detailed non-functional requirements:

- Performance
- Security
- Maintenance / Support

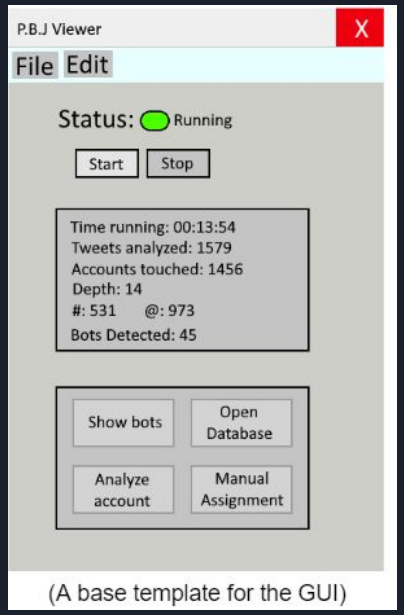
Design Document:

- Created a conceptual model of our Police Bot Framework
- Created a UML diagram for our Police Bot Framework
- Added a detailed description of each module in the UML diagram
- Created an exemplified pseudocode description of the framework
- Created a mock-up of what our Graphical User Interface (GUI) for our framework could look like.

UML Diagram

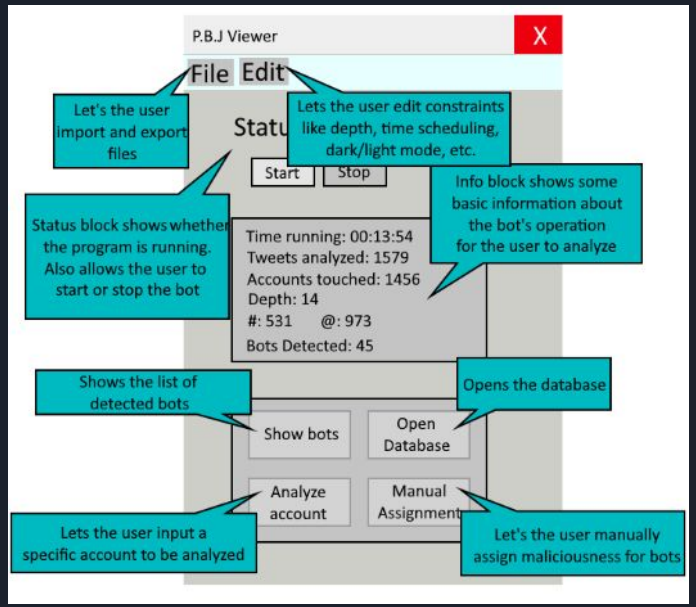

BEGIN
Police Bot Controller
Core Commands:
+Create Police Bot
+Begin Scanning
+Terminate Police Bot
+Detect if Bot
Bot Distinguishing Process
Core Commands:
+Categorize Bot Account Data
+Generate Degree of Maliciousness
+Output Account Info + Decision
Bot Detection Process
Core Commands:
+Model Data
+Apply Detection Algorithm
+Generate likelihood factor
+Return prediction
Deploy Police Bot
Core Commands:
+Create Police Bot
+Link Account
+Terminate Bot
+Define Schedule
Scheduler
Core Commands:
+Create Schedule
Scanning
Core Commands:
+Find New Tweet
+Parse Text
+Pull Account Data
Bot Decision Process
+Generate Recommended Report Action
+Add Account Data to list of Malicious Bots
+Display Malicious Bot List
+Export Malicious Bot List
calls back to
calls back to
calls back to
calls
extends
extends
extends
calls
calls
calls back to

GUI



(A base template for the GUI)

GUI Description



Exemplified Pseudocode

```
Global bot_list;

Main() {
   // method to create a bot using the Twitter/X API / Tweepy
   Bot = new setup_bot(account_credentials, schedule, depth);
   While current_time != bot.schedule_time:
      wait()
      // method that finds the new trending topics and returns them
   Trending_topics = find_trending_topics(bot);
   For each root_tweet in trending_topic:
      scan_tweet(root_tweet, bot.depth, 0);
}
scan_tweet(tweet, max_depth, depth) {
   // limiting the depth of the search
   If depth == max_depth:
      return;
   Account = tweet.account;
   Content = tweet.content;
   Responses[] = tweet.responses;
   // analyzing if the bot is suspicious (still to be determined)
   // and adding it to the list of possible bots
   Suspicious = analyze_suspiciousness(account, content);
   If suspicious:
      bot_list.append(account);
   // repeating the process for the responses
   If len(responses) > 0:
      For each t in responses:
         scan_tweet(t, max_depth, depth+1);

   return;
}
```

Test Plan:

Decided on methods and created testing strategies for the following:

- Detect Function
- Distinguish Function
- Decide Function

Each of these Functionalities will go through the following where the expected result is predefined in terms of output accuracy:

- Unit Testing
- Integration Testing
- System Testing

Milestone 1 Progress Evaluation:

The following was thoroughly detailed for our client who overviewed the Progress Evaluation document along with the others:

- What progress was made on each task (Selecting Technical Tool, Creating “Hello World Demo”, Creation of Requirement, Design, Test Plan documents)
- How each member contributed to each of these tasks
- Discussion on what tasks need to be completed for Milestone 2
- Outlining Client feedback and future meeting times
- Creation of a Task Matrix for Milestone 1 and 2

Milestone 1

Task	Completion	Cody	Gabriel	Liam	To Do
Compare and Select Social Media Tools	50%	10%	30%	10%	Find our social media platform, in case Twitter doesn't work.
Small Demos	60%	5%	50%	5%	Finish working on the technical tools, make sure we want to lock in a specific tool
Compare Collaboration Tools	100%	33%	33%	33%	
Learn the basics of the API	60%	20%	20%	20%	Extend our knowledge further for the twitter API.
Requirement Document	100%	40%	20%	40%	
Design Document	100%	10%	25%	65%	
Test Plan	100%	90%	5%	5%	

Milestone 2

Task	Cody	Gabriel	Liam
Research as many social media APIs as possible (with the possibility of switching from twitter if it becomes unfeasible)	40%	30%	30%
Develop a system to collect basic data on social media accounts	30%	40%	30%
Research known bot detection methods	33%	33%	33%
Research and potentially find a way to store the data we collect	30%	30%	40%

“Hello World” Demo for Tweepy Technical Tool

https://www.youtube.com/watch?v=lRa18QOQcg8

Technical Challenges Update:

Progress on resolving challenges:

- Gained rudimentary experience working with the Twitter API
- Gained rudimentary experience working with the Twitter Virtual Environments
- Gained rudimentary experience working with and coding Bots
- Gained rudimentary experience working with the Tweepy Library
- Expanded HTML knowledge

New Technical Challenges:

- The free plan for the Twitter API does not let users search, scan or reply to Tweets (paid subscriptions allow this functionality but are $100 / Month). This significantly complicates our project plan and means we may have to look into shifting our focus to Reddit, Facebook or Instagram if we cannot get funding from the university.
- Little to no knowledge using Reddit, Facebook and Instagram APIs
- Research must be done to resolve this issue

Moving Towards Milestone 2:

- Resolve Twitter API issue or shift focus to another social media platform
- Conduct extensive research on various tools that are available for other social media platforms (Reddit, Facebook, Instagram)
- Create efficient systems within our framework that can interact and retrieve all relevant data from social media accounts.
- Review academic research provided by Dr. Slhoub on detecting bot accounts and decide on a method / methods of detection.
- Create an efficient account data storage solution, either by creating a locally hosted database or by using AWS
- Ensure we stay within GDPR (General Data Protection Regulation)
- Create a small Demo of our current account data retrieval and storage method

This concludes our presentation, Thank You