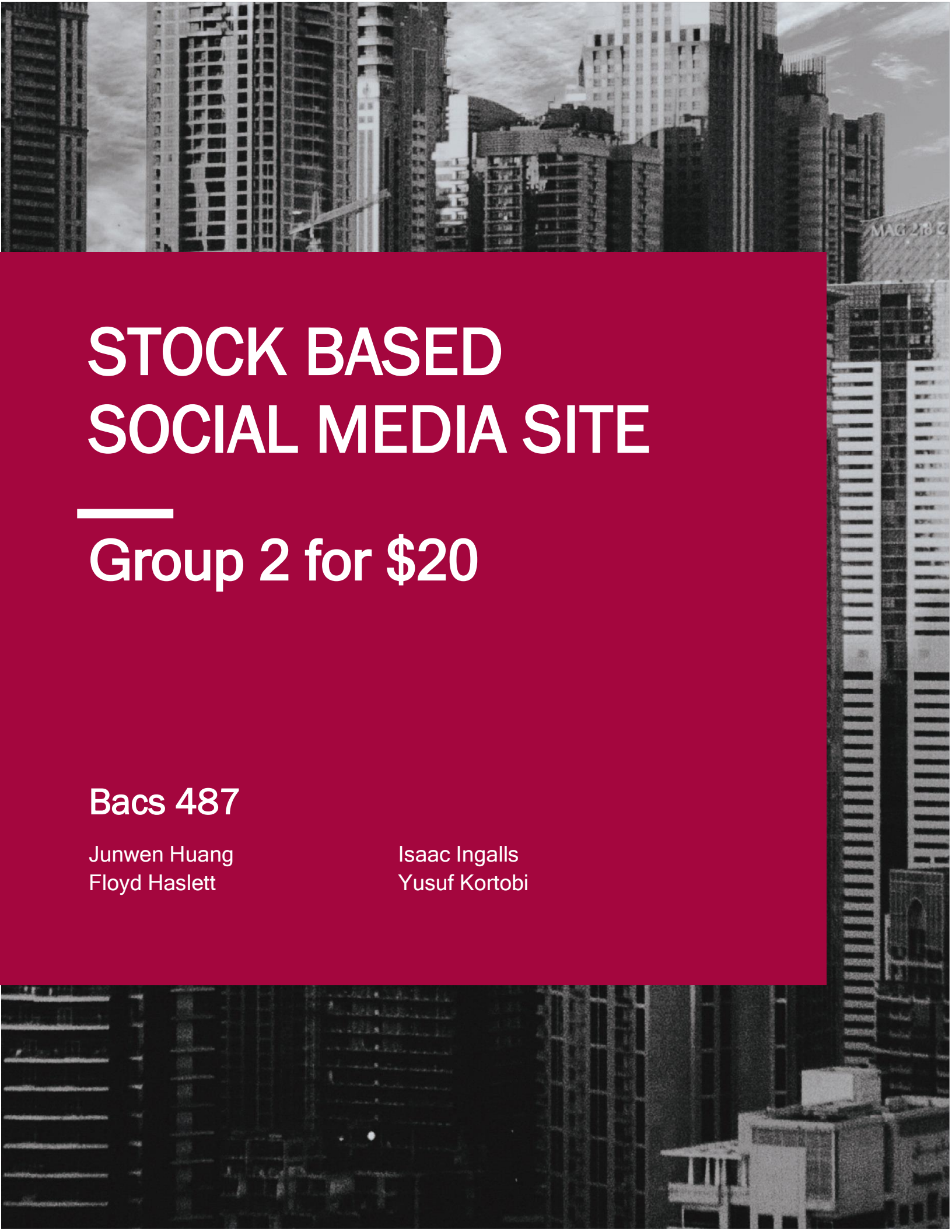


STOCK BASED SOCIAL MEDIA SITE

Group 2 for $20

Bacs 487

Junwen Huang
Floyd Haslett
Isaac Ingalls
Yusuf Kortobi

Planning and Project Management

Roles

- Floyd Haslett
    - Security and Networking Analyst
    - Floyd is responsible for ensuring the security of our website. Making it resistant to attacks and instilling confidence in our users. He is also responsible for the network that our service runs on. Making sure it is robust and reliable.
- Isaac Ingalls
    - Project Manager and Business analyst
    - Isaac is responsible for the business analysis and overall project coordination. It is his job to ensure things are moving on pace and that any issues are addressed. He is also responsible for modeling our responsibilities.
- Yusuf Kortobi
    - DBA and Web developer
    - Yusuf oversees our web development and database capabilities. He will work closely with our UX designer to guarantee that users have an easy time navigating the site and are able to fully engage with all its features.
- Junwen Huang
    - Software and User experience designer
    - Junwen will develop the software functionality to align with our business needs as well as cultivating a user experience that will seamlessly guide users across the platform.

Project / System Request

SPONSOR

In this project, we are going to design and develop a stock trading social network. Our sponsor is Jared Wolfe, who represents the people who would be using the service. Jared is an avid investor, who likes to engage with others on the internet and discuss the latest stock news. We plan on providing the core

services of profiles, the ability to post, and connections to other stock trading websites. Each of these services will be broken down into smaller more manageable goals, but those core goals should make our website run as intended.

NEED

We are trying to solve a problem where retail investors don't have a specific platform to discuss trading. Sites like reddit have formed groups that try to accomplish this goal but are unable to post live analysis with instant integration to other posts. We want to create a community of investors that can communicate with embedded charts and graphs.

REQUIREMENTS

This project is being undertaken to capitalize on an underserved market in the retail investment world. Because of this, users must be able to:

- Get close to real time stock data
- Be able to make posts and interact with other people's posts
- Include charts/graphs in their posts
- Feel secure about their data
- Have a straightforward User experience

VALUE

There is a lot of opportunity in the market for a social media platform centered around the stock market. The ability to post recent stock trends is extremely valuable to both experienced day traders, and first-time investors. We have seen the possible market space for a product like this from reddit users with the explosion of GameStop stock. By creating an avenue for users to discuss stock prices and research stock trends, we would be filling a void in the market.

CONSTRAINTS

Looking at some of the already existing social media platforms, as well as our team's level of experience and familiarity with similar products, the project seems quite doable. However, there are a few constraints that will make the project a bit harder. There are projects that use message boards and social media, and there are projects that use stock information and analyze it for better investments. There are none that do both and finding an effective and compelling way to apply them to each other is the main constraint of this project.

LEVEL OF EFFORT

Because there are four of us in our group, the project is quite complex. Ideally, each of us will invest around 9 to 12 hours in the planning stage since our group have a weekly meeting to talk about how the project should lead, in the analysis and design stage each of us will put in around 10 to 15 hours each week to analyze and design things such as user experience, UI/UX etc. Finally, in the implementation stage, each of us will put in 12 to 15 hours each week to develop all the functionality and do testing, debug etc.

Feasibility analysis

TECHNICAL

For our project, we will be designing a social media platform about the stock market. It will allow people to search by ticker and find independent analysis written by the users of the site. Because our group already has some experienced with technologies such as Java, Python, Django to frame a website and a server etc. However, since some of us in the group do not know as much about stock as the other which means when it comes to development, some of us might find it difficult to develop some of the functionality in our application. Overall, this is a challenging project for us, especially when it comes to implementing numerical analysis and the ability to retrieve live stock data etc. in our application.

ECONOMIC

We are estimating total cost will be around $22,400 if we run this application for about 3 years. Because we do not have an actual client or a sponsor, we are not investing anything other than time into this project. We valued our time at the average junior developer rate of $35 an hour and this project should take us about 640 hours. This will generate an ROI of 346% if we are able to generate our projected $100,000. We came to this projection because we would run ads on our platform and partner with stock trading platforms. With our goal userbase of 10,000 people after 3 years this would give us a million page visits, generating $10,000 of ad revenue, as well as plenty of partnership revenue. Our break even point would be the $22400 but we should be well in excess of that by the end of the project.

ORGANIZATION

Because this is our first time building an application in a stimulating actual work environment, this project has comparably high risk. However, this project is approved by professor Matt Swaffer, and all four of us in our group have agreed that this is a fairly challenging and interesting project to work on as our senior year project. In customers perspective, we will provide a user friendly interface and a system for our user able to login and store their information securely.

Project Plan

APPROACH

We will use a Waterfall based approach with agile elements mixed in. We selected a waterfall approach because with this being our first large scale software development project it seemed wise to lay out as much as we could ahead of time. However, when in the implementation stage we do expect to allocate tasks in an agile style. This will help us get ready for the working world, and it is a natural fit for our group because we will hold weekly meeting to talk about how the sprints went.

TASKS

We will use a WBS to assign how things need to get done. Dependencies will play a large part in how we divvy up which items take priority. We will focus on the stock integration first and build the social media aspects once it is reliable.

TIMELINE

The project is set to take six months and because of our set team size, scope will be the one variable at play. Our preliminary estimates say this project should take about 640 hours to get a working prototype. Over a 16-week semester this is about 10 hours per week per member. We will make our weekly sprints align with this time estimate.

Analysis

We are developing a stock based social media platform where retail investors can come together and share ideas. As a result of this specific target market our platform will need to satisfy specific requirements.

Requirement's definition statement

BUSINESS REQUIREMENTS

- The software must be able to run independent of administration.
- We need to be able to moderate content to ensure it aligns with our company image.
- Software must bring attention to the business

USER REQUIREMENTS

- Find posts based on ticker.
- Make an account
- Make posts
- Engage with other people's posts
- Include dynamic content in posts(I.E charts and drawings.)
- Change account settings (Choose who can see posts)

FUNCTIONAL REQUIREMENTS

Process:

- System must transfer data from a stock api to the website
- System must store user login info/preferences
- System must allow users to share content
- System must allow desktop access (not concerned with mobile access)

Information:

- User info/preferences

- Stock data
- Charts
- Top stories of the day
- Top performers/Failures

NON-FUNCTIONAL REQUIREMENTS

Operational:

- The software will be hosted on the cloud
- It will be written in html with python scripts running backend and sql database storage

Performance:

- It will be designed for 99% yearly uptime
- It will be designed for 100 concurrent users. (people pinging the site within the past 30 minutes)
- Users should be able to access the site on a 1mb/s internet speed. Seeing posts within % minutes of their upload by the other user.

Security

- The public will see the front end of the system
- We will be medium security as we offer little personal identifying info(no credit cards)
- Backend will only be available to admin(not open source)
- The service will encrypt usernames/passwords but its security will be in that most the info is publicly available making us a weak target.

Culture and Political:

- The system will be set up in America to talk about the American stock exchange.
- We will have to abide by all sec regulations regarding insider trading. I.e., limiting what companies people can talk about depending on their involvement with that firm.

- Other than that users can say what they want and it falls on us to deem what content is acceptable beyond insider trading.
- Culturally we must encourage good investment strategies.(This may include filtering out posts of the YOLO variety)

User Stories

I. As a public user, I want to create an account (Epic)

II. As a public user, I want to link my account to other forms of social media

   a. Acceptance criteria: link account to the required email address with optional social media

III. As a public user, I want to personalize my account

   a. Acceptance criteria: personalization for a photo, username, screen name, and password change

IV. As a user, I want to make my account secure

   a. Acceptance criteria: password requirement criteria, 2fa, privacy settings, and encrypted data

V. As a public user, I want to report technical issue I run into

   a. Acceptance criteria: A customer support page for users to submit a ticket to report technical issue they have

VI. As a public user, I want to be able to select a “forgot password” button and go through a password recovery or change path.

VII. As a public user, I want to be able to log out of my account

   a. Acceptance criteria: views must exist for not logged in users

VIII. As a public user, I want to be able to delete my account

   a. Acceptance criteria: option to remove the user from a back-end database

IX. As a public user, I want to have a say in who can see my account and posts

a. Acceptance criteria: Privacy settings on the account can be set to allow everyone to see the recent history, and when publishing a post, you can set the post to be viewable by everybody or have select viewing options.

X. As a public user, I want to be able to switch from one account to another

a. Acceptance criteria: adding a switch account function under the logout function for users that have multiple accounts to switch

XI. As a public user, I want to publish my own stock analysis feeds

a. Acceptance criteria: The user should be able to post a description of their investment, or investment strategy, in the form of text and pictures.

XII. As a public user, I want to able to delete my own post

a. Acceptance criteria: having a delete function for any posts that users want to delete

XIII. As a public user, I want to able to go back to edit posts that have already publish

a. Acceptance criteria: having an editing function under the users post for them to go back to edit

XIV. As a public user, I want to view other people's feeds

a. Acceptance criteria: save information about the user as posts on their account, allow for privacy settings by the user for who can view their recent activity outside of posts.

XV. As a public user, I want to be able to view news about the most current stock market.

a. Acceptance criteria: by importing stock information, the user should be able to sort by the biggest and lowest % stocks, as well as view posts made about that stock made by users.

XVI. As a public user, I want to view the change of a specific stock in a certain time interval(I'm thinking like the past 3 years as maximum?)

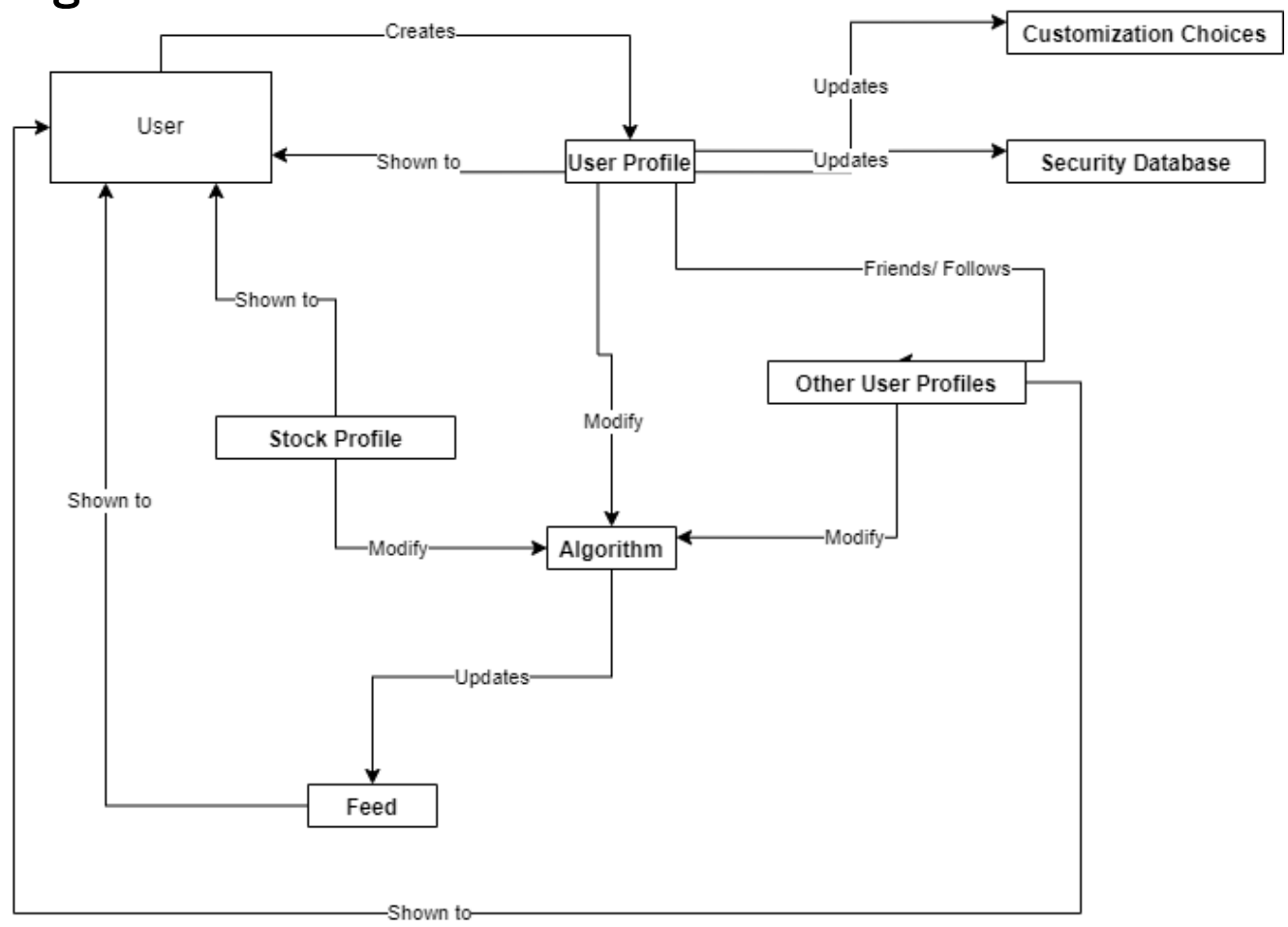
a. Acceptance criteria: Access to stock pricing information, up to the creation of the stock in the market, and allow for the user to choose the interval the user wants to see

XVII. As a public user, I want to view what kind of stocks are the most trending to invest

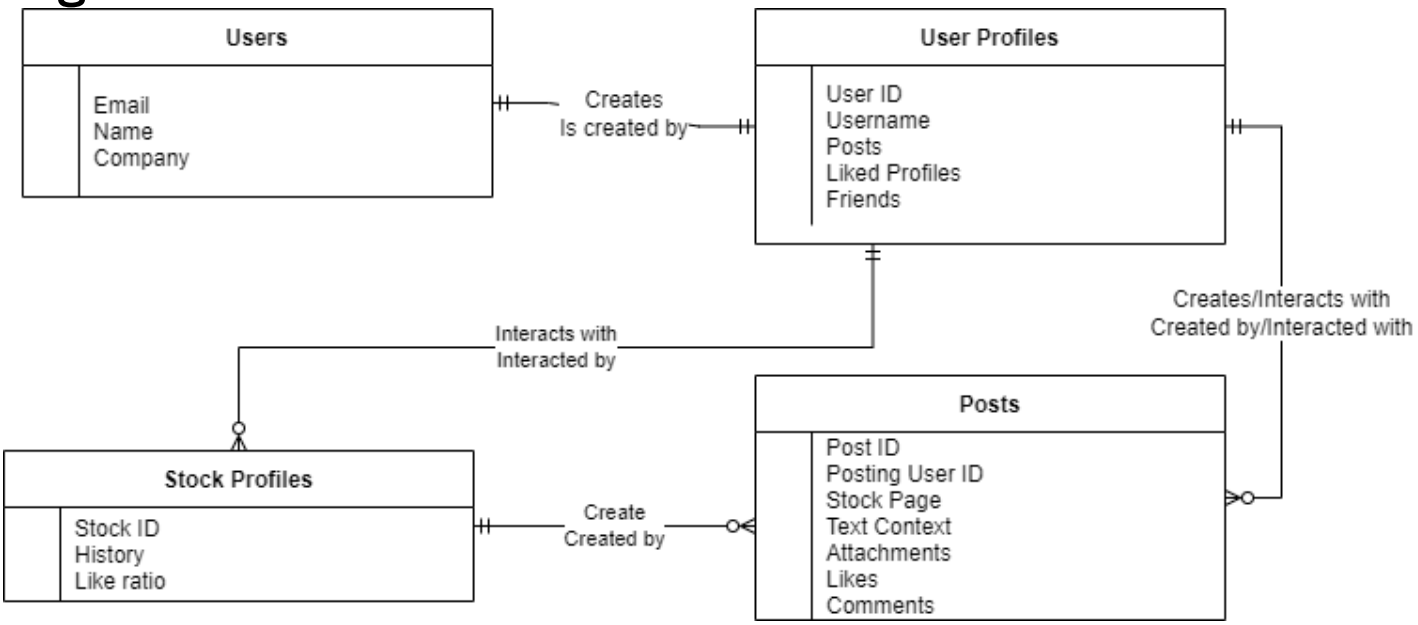
a. Acceptance criteria: A single view could be used to show posts, and the user should be able to click on the post title to view their post, and click on the user to view the user's past history depending on user privacy settings.

XVIII. As a public user, I want to see my own post history

Logical DFD



Logical ERD


Users
Email
Name
Company
Creates
Is created by
User Profiles
User ID
Username
Posts
Liked Profiles
Friends
Creates/Interacts with
Created by/Interacted with
Interacts with
Interacted by
Posts
Post ID
Posting User ID
Stock Page
Text Context
Attachments
Likes
Comments
Stock Profiles
Stock ID
History
Like ratio
Create
Created by

Design

Design narrative

REQUIREMENTS ACQUISITION

Requirements were gathered via a direct user interview as well as consultation with relevant documents. These requirements were then translated into our user stories which will be used to direct the projects development. These requirements are not static and will change during the process of iterative software development. The requirements we have now serve as a baseline for our project, but we do anticipate some new requirements to come up throughout the process of building the software. Due to this, we have made it a point to bring up any new issues during the weekly meeting we will be holding.

TASK ALLOCATION

Tasks will be allocated based on user stories, with the goal of each set of tasks being completed within a one-week sprint. We chose this task allocation structure because it allows the most accountability as well as an even distribution of workload. If one member consistently finishes all their tasks in under one weeks' time while another struggles to make the deadlines there is an easy solution to make a more equitable distribution of tasks. Members will be responsible for completing all user stories within a one-week sprint. If a member is consistently unable to finish their projects within a one-week sprint they will have to explain why during the team meetings so we can find a way to improve their completion rate. User stories will be divided up based on familiarity first, then based on necessity, and finally they will be divided up based on desirability. For example, someone with a strong familiarity with UI design will be tasked with designing a homepage, while someone with a better database background would oversee data storage/acquisition. But if they had completed their relative field but there was still no user login system, that would be assigned to them next as it is a necessary part of the project. Finally if most of the site is built and functioning but there are optional features that could be implemented we will divvy up those responsibilities based on how much each member wants to do them. User stories will not be marked as completed until they are implemented in the working build and functioning in accordance with

their acceptance criteria. Each member will also have to write an appropriate section of a user guide as they work on user stories.

ACCOUNTABILITY

Members will be required to attend one weekly meeting where we discuss our current progress and allocate the next batch of user stories. This meeting will be paramount to the success of our project because each person will be able to share any experiences/troubles they had while building the site. Enabling us to all be on the same page and deliver a better product as a result. If a member is unable to make the meeting, they must give the others 24 hours of notice beforehand. It will not be held against them if their user stories are completed but if they are not that member will have to provide an adequate explanation during the next meeting or face some consequences.

TESTING

The product will be iteratively tested in accordance with the acceptance criteria for each user story. This will mainly be user and unit testing conducted by the person writing the code as well as a small group of beta testers who were picked based on a general familiarity with the subject matter. Beta testing will not occur until most of the site is complete (determined by how many epic user stories we have completed). Once we reach that point beta testers will try to act as if they were trying to use the site normally, and after it passes those tests/feedback we will instruct them to try to find bugs by doing extreme things.

List of Technologies

Frontend technologies and programming languages: notepad++, visual studio code, HTML, CSS, JavaScript, Bootstrap, React

We chose most of these coding languages based on previous knowledge, familiarity, and understanding for the more complex aspects of this project. Notepad++ and visual studio code are our preferred method of writing, as they give the user error messages and allow the user to fix and modify their code with ease. HTML CSS and JavaScript are all a packaged deal, with how to make a website look pretty, and because we want it to look professional we will likely add some bootstrap to the pages. React will be used more in the background to make the web app work as efficiently as it can while also elevating the common user experience.

Backend technologies and programming languages: Python Django

Django is a way to create web applications that we have a good amount of experience in. Django uses python to link pages together, and link them to a hosting service. The python being used will also be our method of data collection, and will be sending the user data we collect to our sqlite server.

Data storage: SQLdeveloper

SQL developer is what the database management course here teaches, and has ample methods to create, manipulate, and store data. SQL is an industry standard for data storage, and while not every member on the team is familiar with the language, there are many guides and examples that can be found to help learn the intricacies of the language.

Web servers: pythonAnywhere

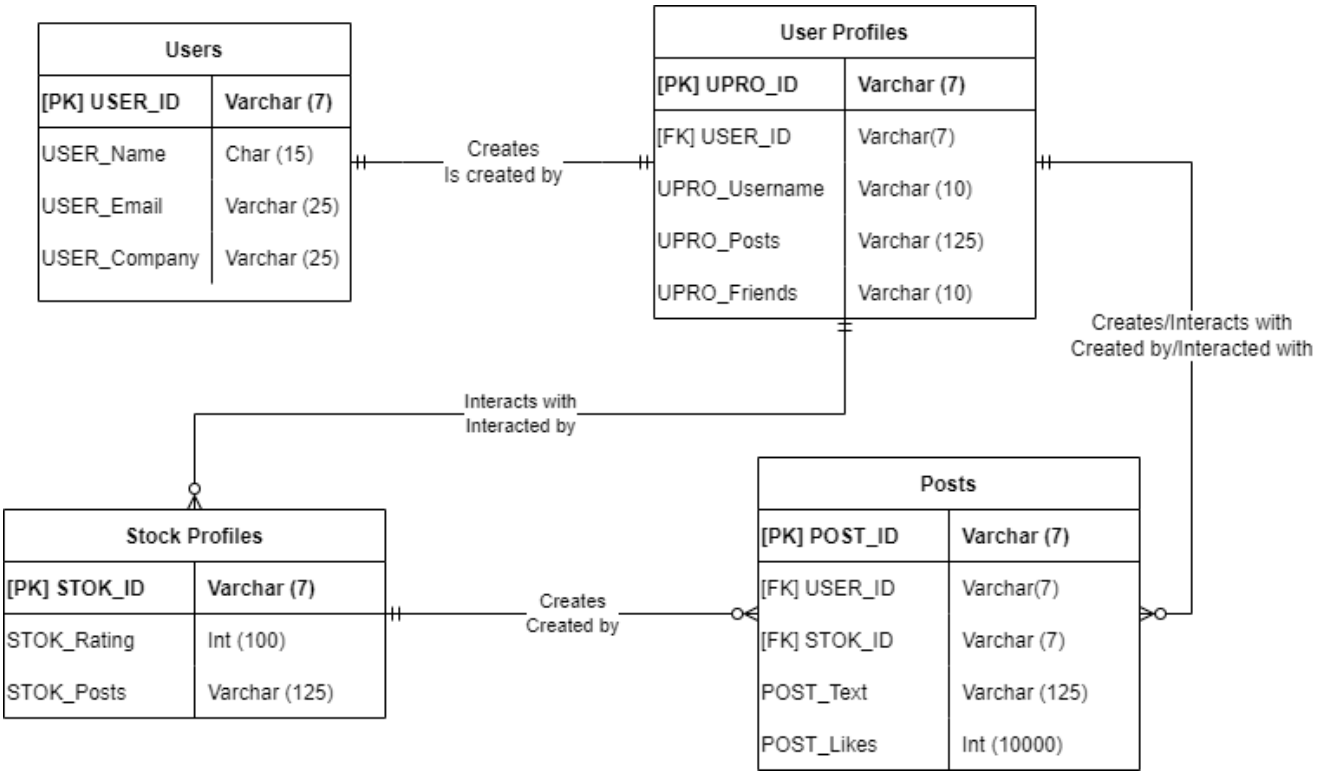
We are unsure if pythonAnywhere will be what we use for the final implementation, but for now we are planning to use it, again, because of past experience using pythonAnywhere in addition to django because they are both free services. The reason we would use a different web hosting service would be to bring a degree of professionalism to the project, and allow us to present it better on resumes.

Version control technology: Github, Github desktop, Git

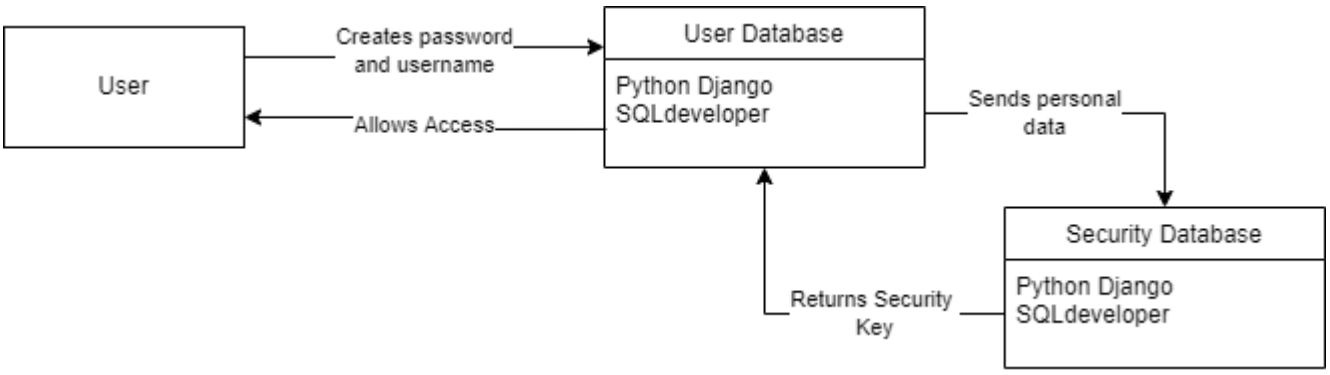
Git Github and Github desktop are also industry standards, and are what most actual developers use to work in teams on projects. Being able to commit and pull will be extremely valuable to our project. While there might be other source control programs out there, our team is more familiar with and prefers to use Github.

Physical Process Models

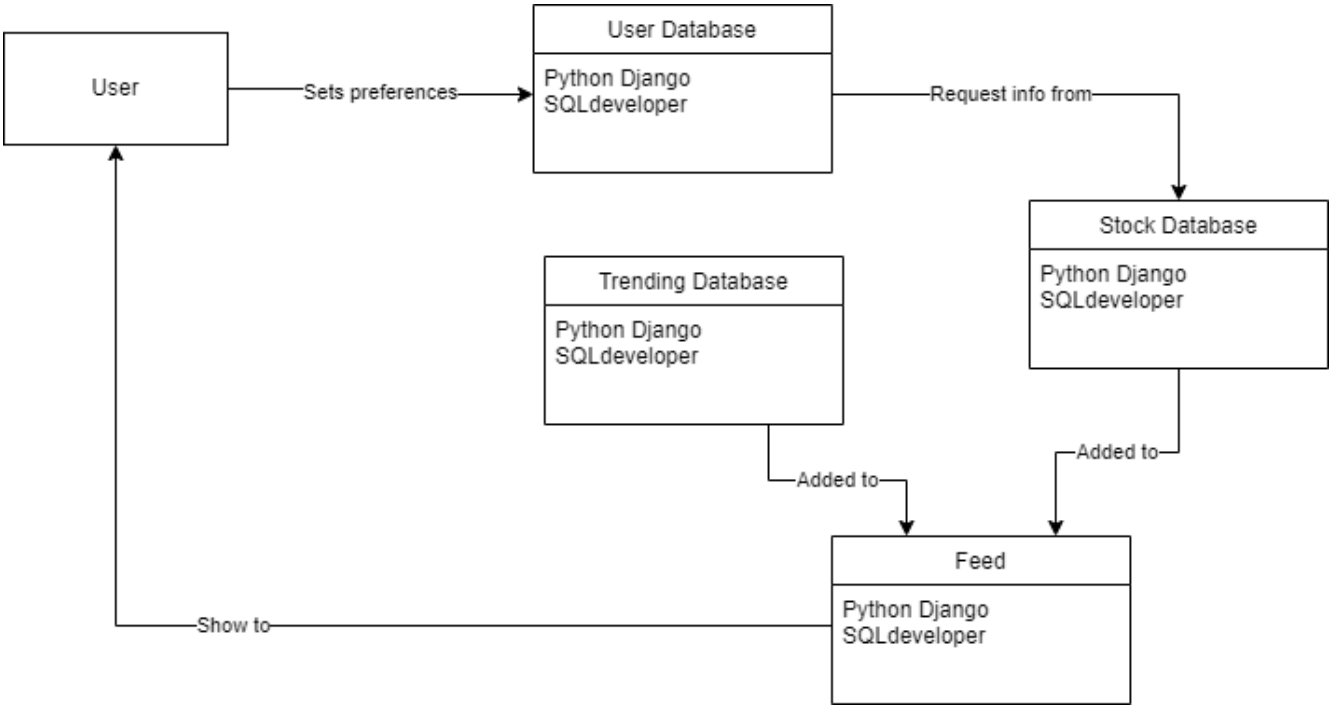
Physical ERD



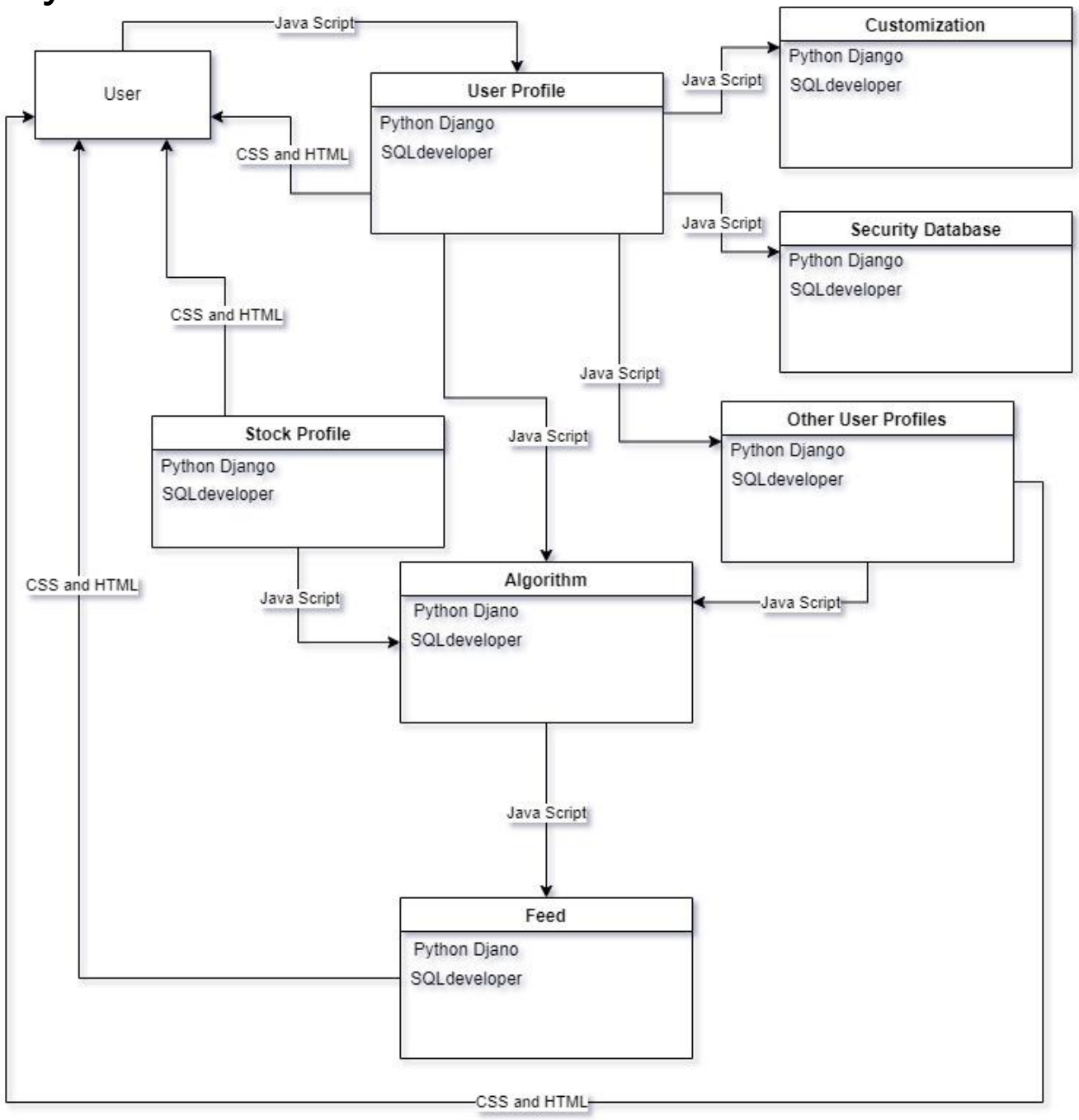
Security Data Model



Home Feed Data Model



Physical Dataflow Model



Implementation

Source Control

The repository for our work on this project can be found at https://github.com/June907/2420. This is a team repository and will allow us to quickly and easily share code with each other. We will be committing our work every time a user story card has been achieved, and we will be using GitHub desktop and git through command prompt to do so. Git is the industry standard for version/source control, and trying to use a different program to share and update code would result in a waste of time becoming acquainted with this method.

Work Assignments

At every weekly meeting, each person in the group will be assigned user stories that they are to develop and implement into the code. We will have target times for completion and will work as a team to build the website with add-ons. The team will hold each other accountable, and we will keep track of who does what through the Trello app. Team members will be assigned user stories based on their familiarity with the subject at hand. Yusuf is the most familiar with database design and will be responsible for a lot of our database architecture. Isaac is most comfortable writing scripts and will therefore be tasked with our data transfers. Floyd is at home with security concepts and will make sure users' data is secured. Junwen is a good UX designer and will oversee that. When tasks fall outside our comfort zones, we will discuss who would like to take that responsibility during the team meetings.

Testing

We will conduct unit tests as the various processes are developed. These tests will be done in accordance with the components acceptance criteria and the test will not be considered "passed" until the process meets all expectations. These unit tests will be written by the same person who developed the process and once it passes those tests other team members will be given the opportunity to write other tests.

Once the process has passed its unit tests it will then be integrated into the system and we will conduct integration tests. This will be done by integrating the new process into its larger group first, and once it is proven to be stable it will be

added to the site as a whole. For example, if someone creates a system to encrypt passwords it will first be added to the database system to make sure it integrates there. Then it would be implemented sitewide to limit the chance of unknown complications.
After the majority of the system is built and has passed its unit and integration tests we will conduct acceptance tests. These tests will be conducted by our team and eventually by a small group of beta testers. These tests will be primarily focused on UX and UI but will serve to find any flaws in the larger system as well.

Documentation

For the system we are implementing next semester, we will be writing up documentation for all the codes that we're going to implement next semester. Documentation will be produced in Visual Studio Code, which is the same editor we will be using for our codes. We will explain the functionality for each portion of the code since it will be more efficient to fix errors that we might run into. We will also have a separate document in our GitHub that explains in detail what our system does after we finish implementing our system, that way it will give our audience a general idea of our product.
In addition, each individual will be documenting codes that they wrote, but at the end of the semester, we will have one person write a complete document for our system for the audience.