Team Syrup

Daniel Pulido, Paul Collins, Nirranjan Akilan, Nikhil Dasari,
Brandon Coffey, Scott Pedley, Monish Ravishankar

1. Project Description

Draft Description: We will design an app to save users money by displaying the price history of an item to help find the lowest cost of a product. This app will use various vendors to compare the price of the desired product, letting the user know if the product is above or below the Manufacturer's Suggested Retail Price (MSRP).

Instructor Feedback: A lovely topic!! Once complete, it will save a lot of time and effort for those who are in dire need to find an item at the best price possible. Please consider implementing it fully, if you can. No pressure w.r.t. grade on implementation. In the final report, please make sure to include comparison with similar applications -if any-, make sure that you differentiate your design from those, and explicitly specify how. Fair delegation of tasks. Please share this feedback with your group members. You are good to go. Have fun with the project and hope everyone enjoys the collaboration.

As per our instructor's feedback, we will research other applications that share a similar niche with our topic to draw inspiration for features to ensure that our application is unique. This will help us get familiar with our "competitors" should we consider making the full implementation.

2. GitHub Repository

https://github.com/BrandonMCoffey/3354-TeamSyrup

3. Teammate Tasks

Nirranjan:

- Major contributor to creating the software requirements.
- Overall contributor to designing the architectural design diagram.
- Responsible for submitting to elearning (including GitHub URL).

Monish:

- Overall contributor to the modeling process, specifically responsible for the Class Diagram.
- Overall contributor to the report, specifically organization and writing.
- Responsible for making an additional commit about project scope .

Daniel:

- Overall contributor to the sequence diagram.

Paul:

- Assist Monish with documentation of report and editing.
- Responsible for facilitating architectural design pattern diagram.

Brandon:

- Create a GitHub Repository and add all team members and the TA as collaborators.
- Assist Daniel with the Sequence Diagram.

Nikhil:

- Making first commit to the repository (README).
- Responsible for creating the Use Case diagram(s).

Scott:

- Software process model contributor.
- Contribute to setting up the GitHub.

4. Software Development Model

Our group has decided to utilize the incremental process model as it allows small teams to set manageable deadlines while keeping our resources and expenditure in check. The incremental process model will allow us to start with simple features that we make more complex as time progresses. Our project's first increment would be retrieving price history information from APIs and store it in our application's backend since price data is the core aspect of our application. Our project's second increment would be designing our application's front end since this is where our users will interact with our service, and our goals are to make it easy to navigate, visually appealing, and support our required features. Our project's third increment would be connecting the front end to the backend so that our website isn't entirely static and actively retrieves & renders price history data from our database.

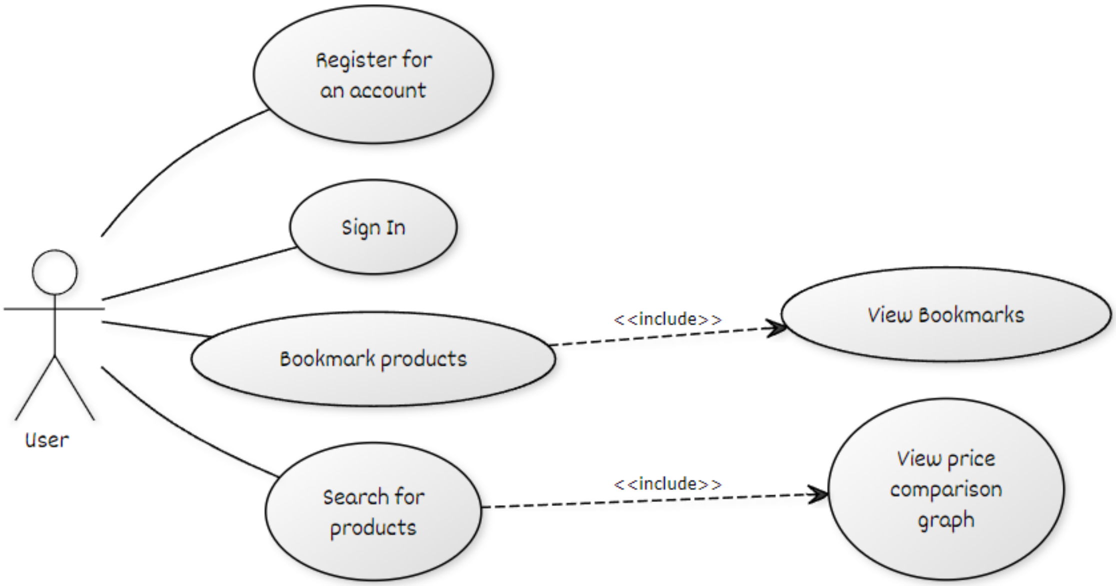
5. Functional Requirements

1. API calls to get product information that we update once a day and track for a set time (ex: a year).
2. A search bar where the user can type in an item and click on it to go to its page
3. A query function to find the cost of the item from all the different shopping websites
4. Have functionality to graph a time vs price graph visual for users to view
5. Functionality to register/login in and save user information
6. Allow users to bookmark products to their account
7. Display bookmarked items on the user's homepage

Non-Functional Requirements

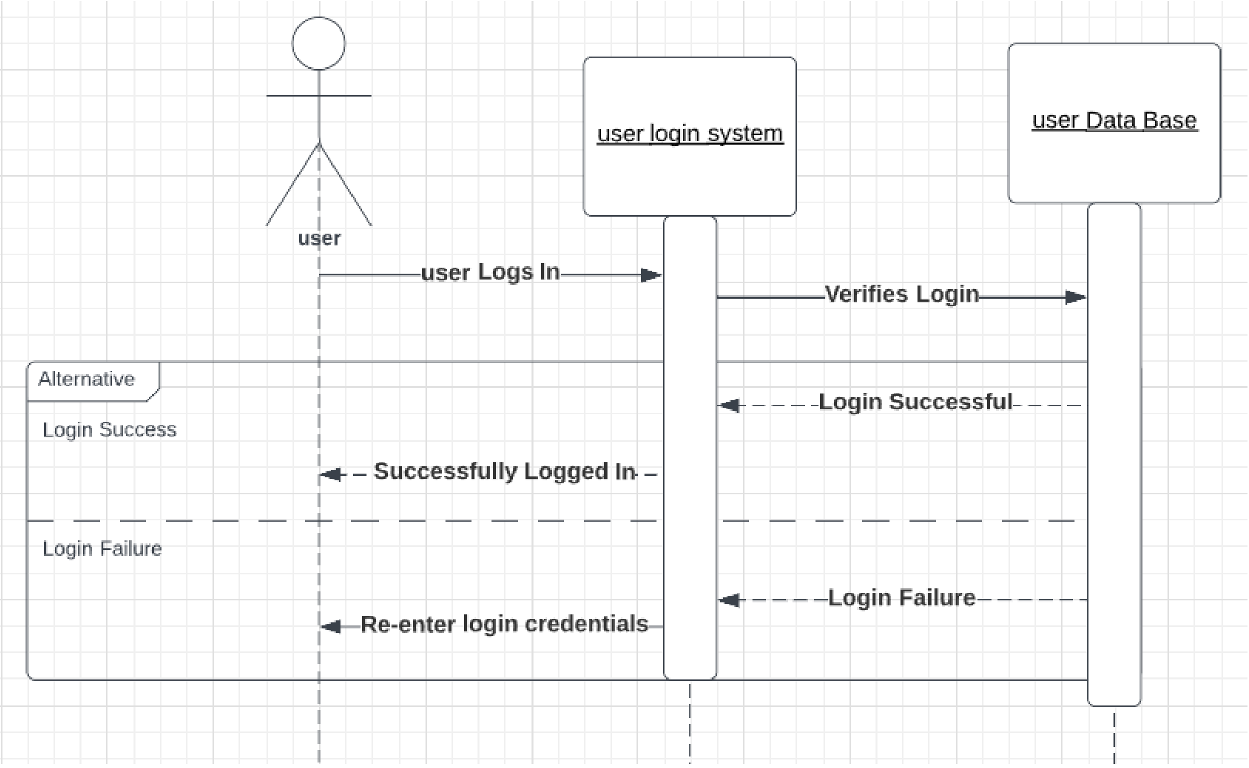
Usability requirement: Should be able to be used on desktop and mobile.
Performance requirement: All requests must be generated in 2 seconds or less.
Space requirement: No data should be stored on the user's computer.
Environmental requirement: The computer hosting the application's server must be located in a dust free environment per American dust laws about server maintenance.
Operational requirement: The application must be able to gather price history once a day and track it for a year.
Development requirement: The application must implement object-oriented programming practices to ensure that price data calculations are abstracted from the user.
Accounting requirement: The application's shareholders get a small commission for products purchased through the application.
Safety requirement: All user and cite data should be encrypted.

6. Use case diagram

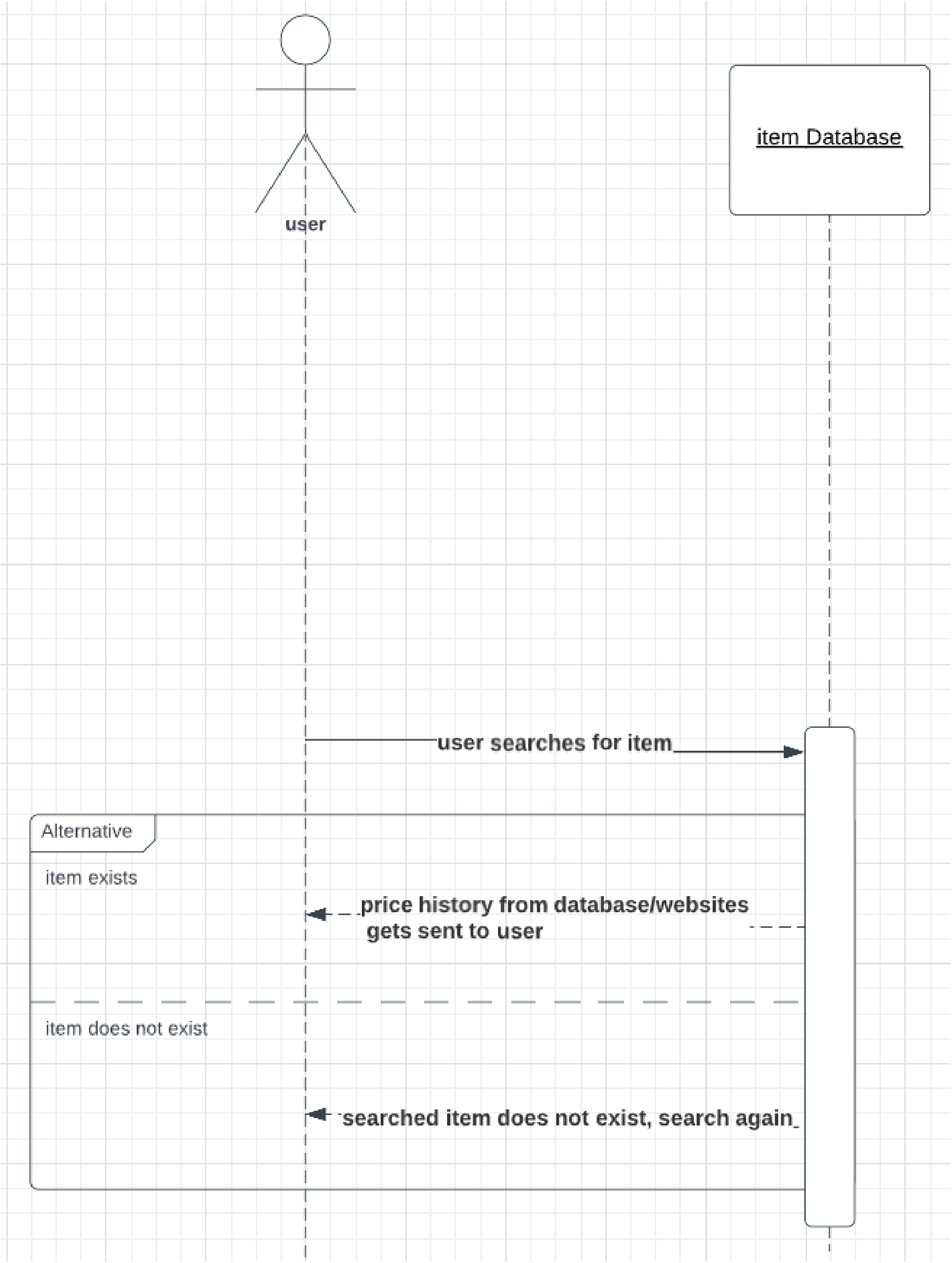


7. Sequence diagram

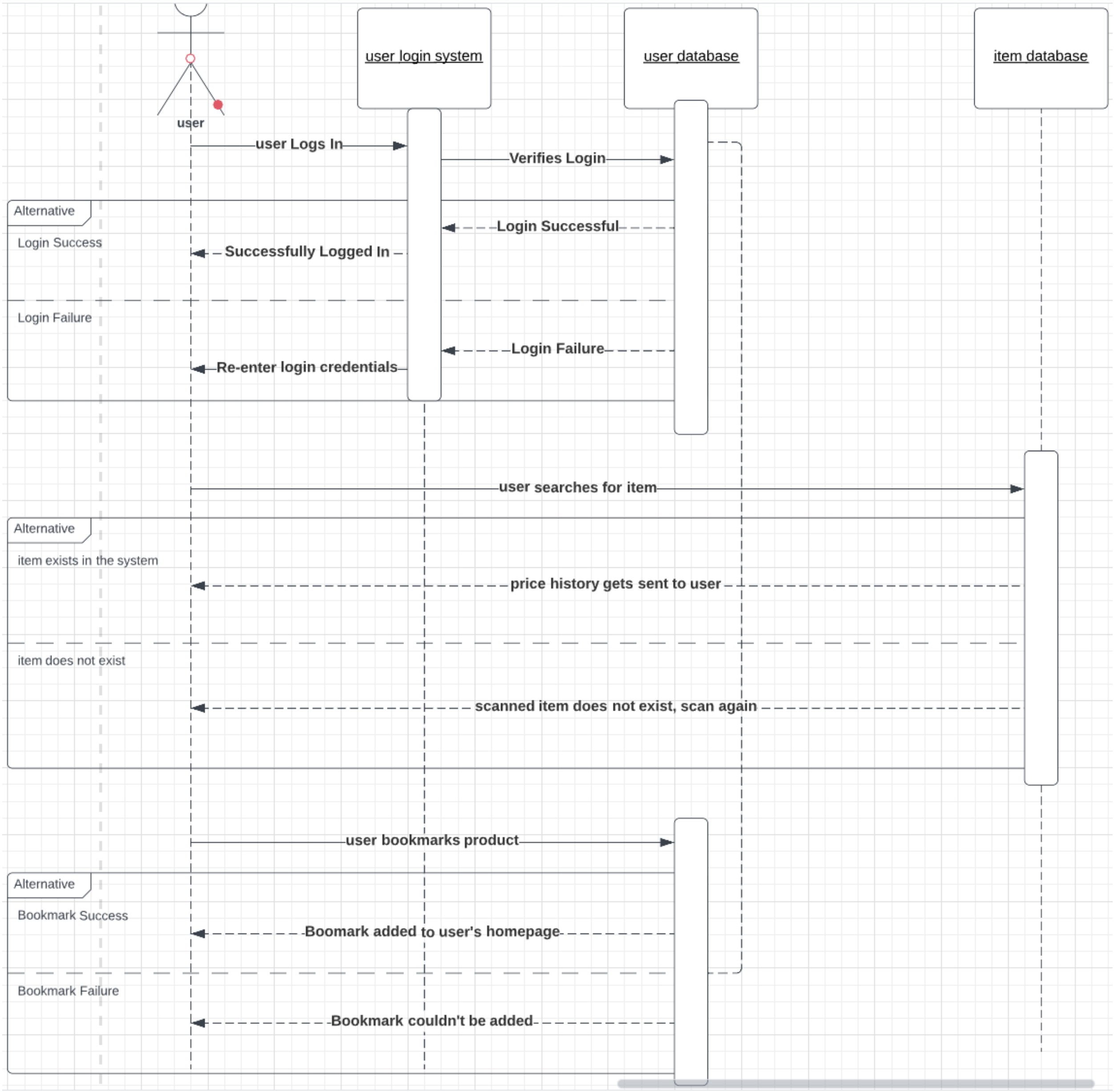
Login/Register:



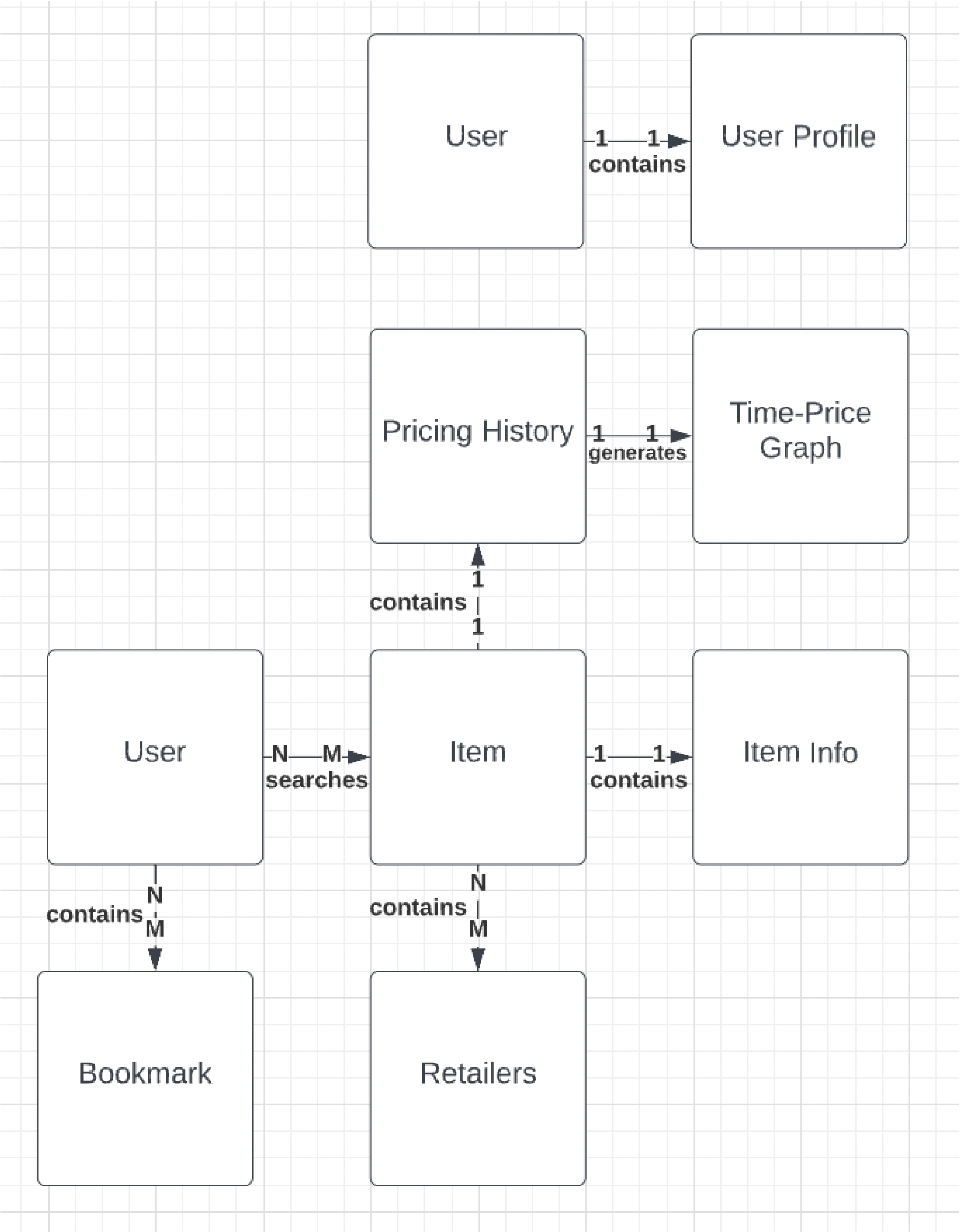
Search for product:

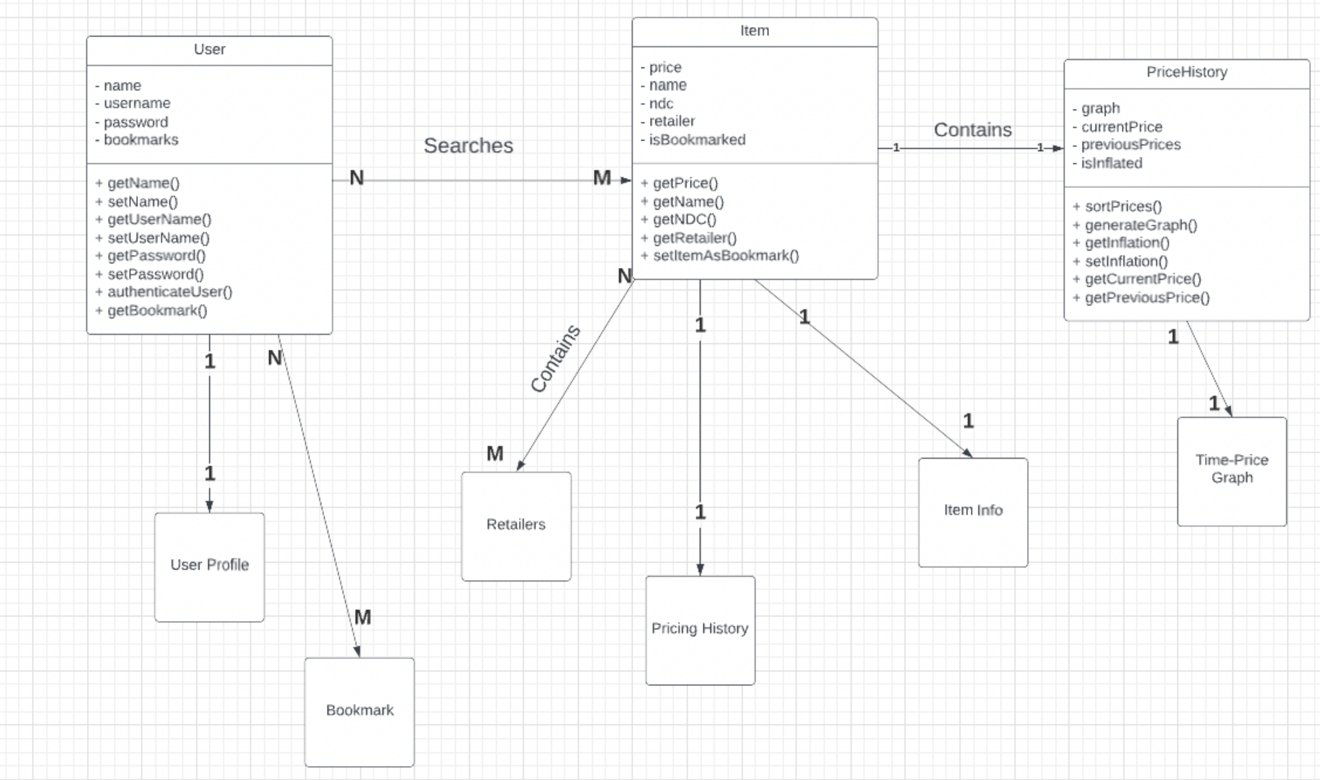


Bookmark Product (need to be logged in and have searched for an item)



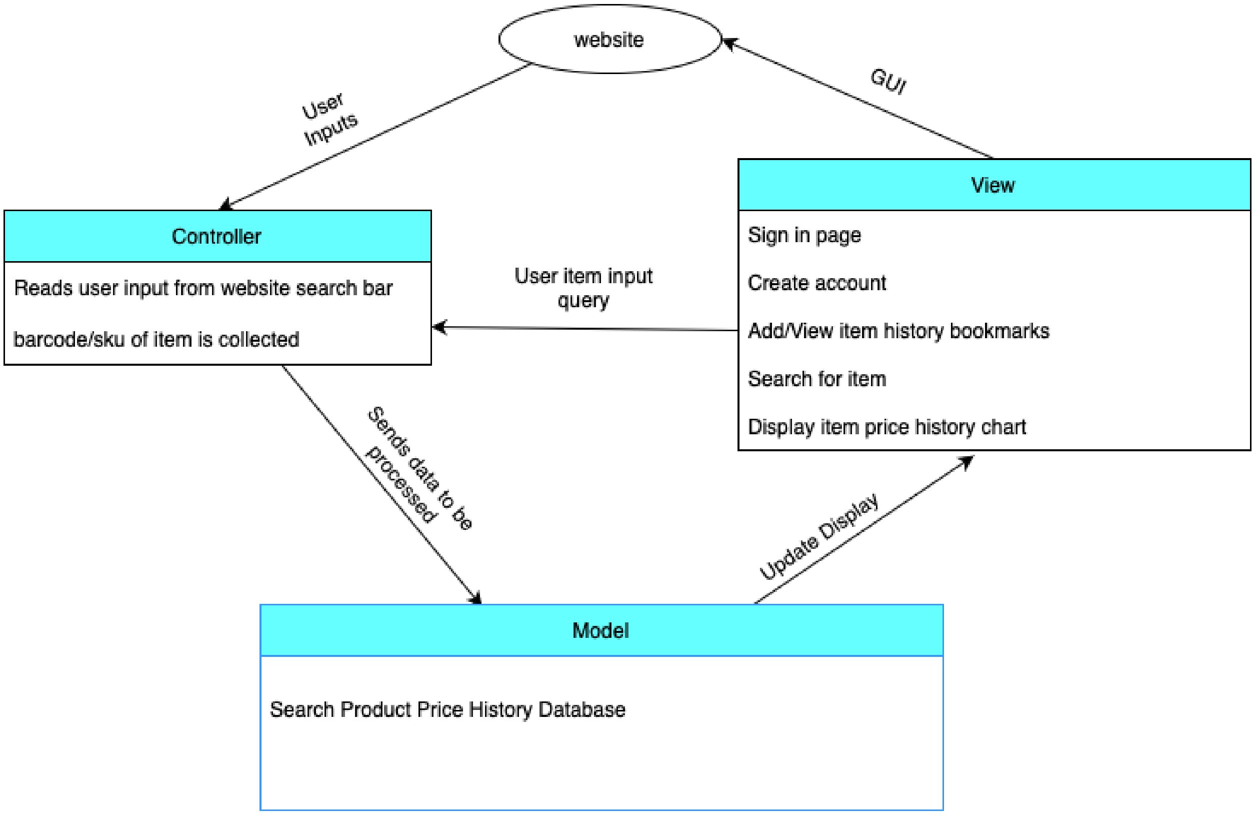
8. Class diagram




User
- name
- username
- password
- bookmarks
+ getName()
+ setName()
+ getUserName()
+ setUserName()
+ getPassword()
+ setPassword()
+ authenticateUser()
+ getBookmark()
Searches
N
M
Item
- price
- name
- ndc
- retailer
- isBookmarked
+ getPrice()
+ getName()
+ getNDC()
+ getRetailer()
+ setItemAsBookmark()
Contains
1
1
PriceHistory
- graph
- currentPrice
- previousPrices
- isInflated
+ sortPrices()
+ generateGraph()
+ getInflation()
+ setInflation()
+ getCurrentPrice()
+ getPreviousPrice()
1
N
1
M
User Profile
Bookmark
N
Contains
M
Retailers
1
1
Pricing History
1
1
Item Info
1
1
Time-Price Graph

9. Architectural design

Model-View-Controller (MVC) pattern


website
User Inputs
GUI
View
Sign in page
Create account
Add/View item history bookmarks
Search for item
Display item price history chart
Controller
Reads user input from website search bar
barcode/sku of item is collected
User item input query
Sends data to be processed
Update Display
Model
Search Product Price History Database