1. **List the features that were implemented (table with ID and title).**

ID	Requirements
BR-1	Catalogs of products are categorized based on hardware, condition, etc.
BR-2	User account must be email format and free of cost
BR-4	Notification email of transactions should be sent within 5 minutes

ID	Requirements
UR-01	As a user, I want to be able to create an account
UR-02	As a user, I want to be able to log in to my account
UR-03	As a user, I want to be able to be able to browse certain types of electronics with a search filter
UR-04	As a user, I want to be able to view product detail information
UR-05	As a user, I want to add product to the inventory
UR-06	As a user, I want to edit product information
UR-07	As a user, I want to remove product from the inventory
UR-08	As a user, I want to add products of interests to my shopping cart

UR-10	As a user, I want to edit my account information including shipping address and payment information
UR-11	As a user, I want an email notification for every exchange I have made

ID	Requirements
NFR-01	Password security authentication
NFR-02	Users shall receive notification of profile changes via email
NFR-03	Product information and wish-lists will be stored in database

2. **List the features were not implemented from Part 2 (table with ID and title).**

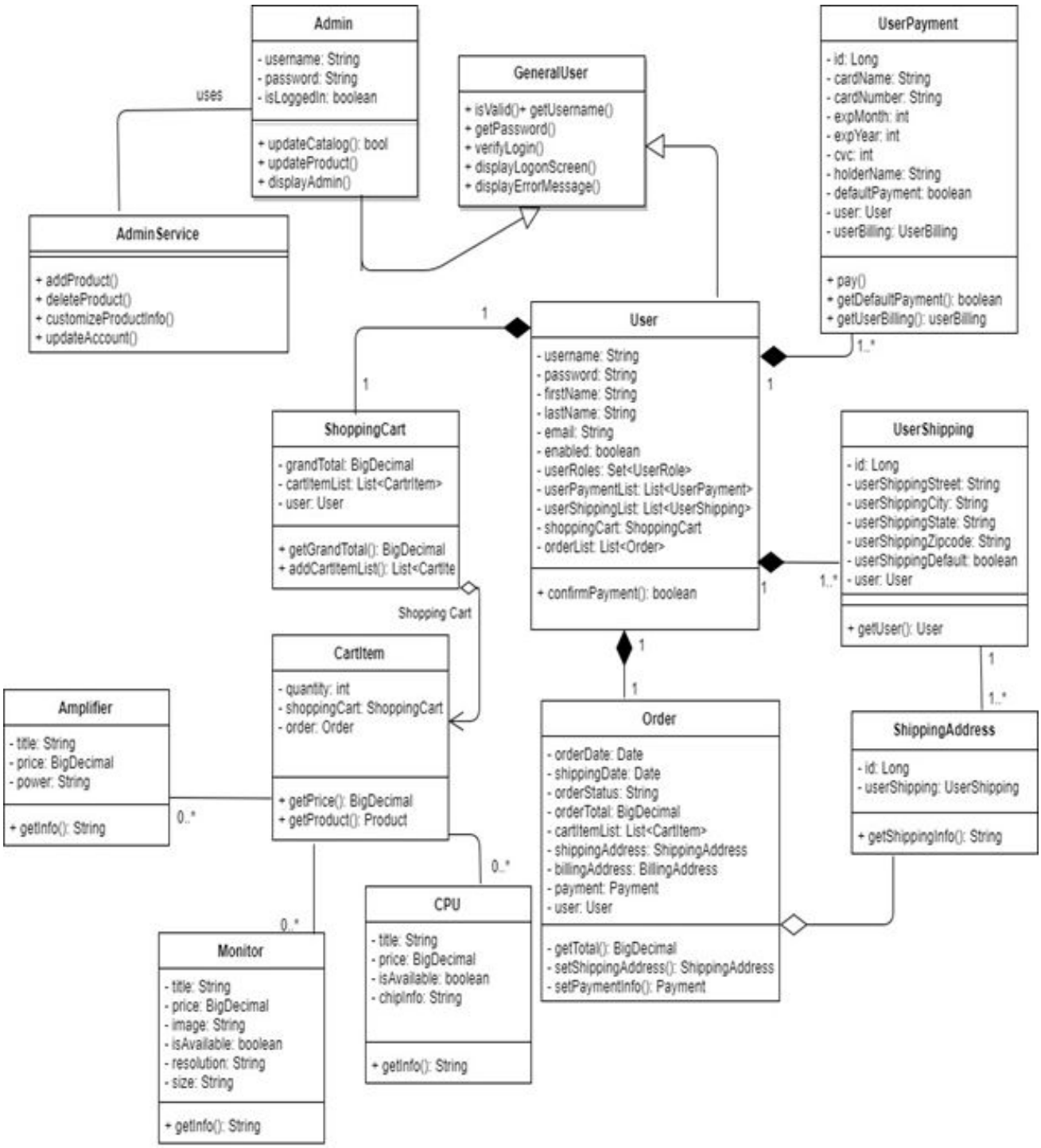
ID	Requirements
BR-3	Advanced product search facility with preferences and filters
BR-5	Shopping cart will be invalidated after a day

ID	Requirements
UR-09	As a user, I want to confirm buying products from the shopping cart

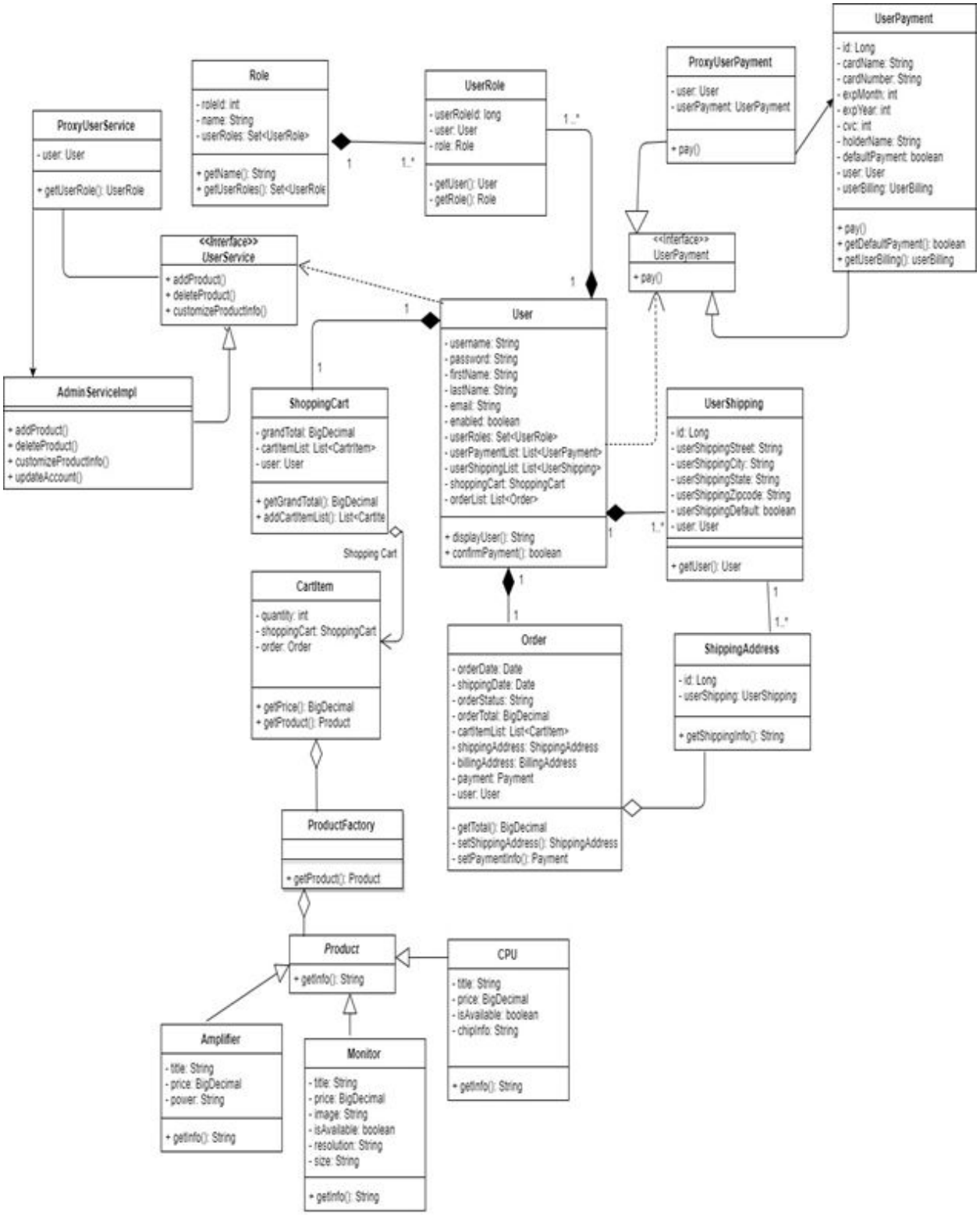
ID	Requirements
NFR-04	Searching engine must be dynamically updated and run fast
NFR-05	Credit card information must be kept private

3. **Show your Part 2 class diagram and your final class diagram. What changed? Why? If it did not change much, then discuss how doing the design up front helped in the development.**

Old Class Diagram(part 2)



New Class Diagram

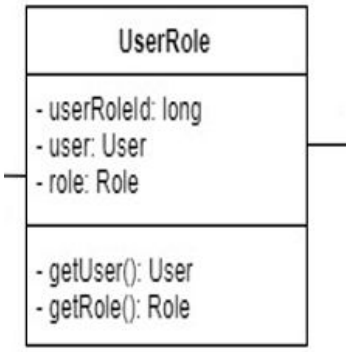


3(cont)

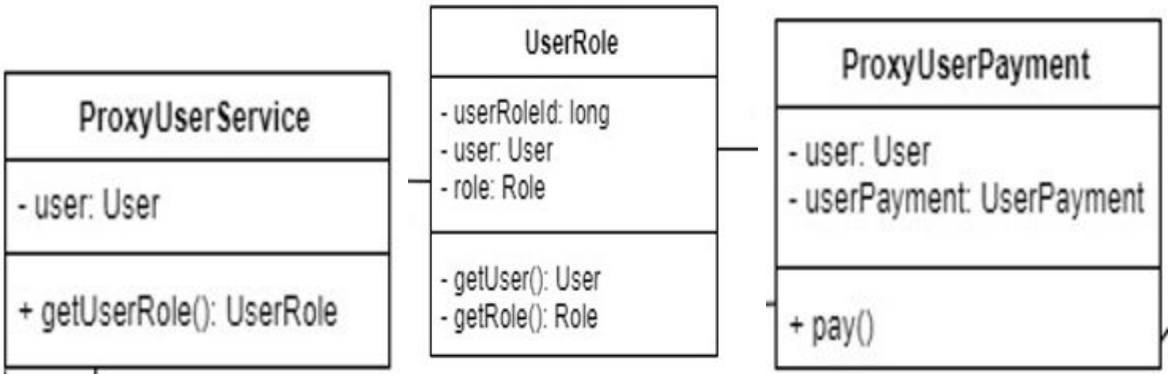
Many of the relationships between the classes had been altered to better fit our new design patterns. We added a product class so we could use inheritance better to allow for numerous different product categories. This was added because of one the the design patterns we implemented. We also added many proxy classes to help check the users roll and see if they are allowed to do certain actions. There is a proxy roll for admin users adding new products and a roll for regular users who are purchasing products that medagates the users purchase information.

4. **Did you make use of any design patterns in the implementation of your final prototype?**
If so, how? Show the classes from your class diagram that implement each design pattern (each design pattern as a separate image in the .PDF).
If not, where could you make use of design patterns in your system? Show a class diagram of how you could implement each design pattern and compare how it would change from your current class diagram.

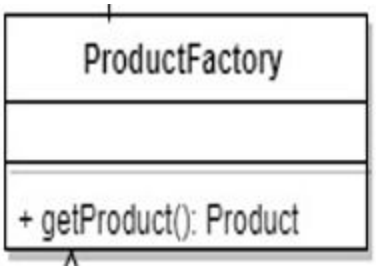
We implemented a Proxy design pattern so that we could work with varying states better. One of the examples is to allow the system to easily identify between a regular user on the site who is browsing and buying products and an admin who is created and selling products.



We also implemented a few classes to help manage the communication between objects.



Another design pattern that was added was the Simple Factory Design Pattern as the ProductFactory object so that we could easily create new object classes with no large effort on the database.



5. **What have you learned about the process of analysis and design now that you have stepped through the process to create, design and implement a system?**

While going through the process of analysis and design, we learned that actually coding the system comes last. This was rather different from other classes where finding a solution by coding is the primary way to solve assignments. Going through an entire design process allowed us to solve problems by looking at the entire system overview instead of focusing on each error message. By looking at a high level overview we could take advantage of design patterns to solve problems ahead of time without wasting time doing needless coding.