Spring 2018 CS2340 Team Final

You will take this final as a team, or part of a team depending on group decisions. You may collaborate freely with your team mates, but may not share information with another team.

Please submit before the end of final exam period (Thursday May 3d 1740) this one page with your team name and members (this page can also be scanned or photo'd and electronically submitted). If you submitted final materials electronically, please indicate who submitted the materials so I can find them. You may have a mixture of hard copy and electronic submission, or may choose all hard copy if you like. Hard copy materials must be turned in by the end of the exam period along with this sheet. You may bring materials to my office (Room 138 CoC). If I am not there, just slide under the door.

For the UML questions, if you use Visual Paradigm (or another tool) please export to an image if at all possible. This will ensure that we can read your work without version conflicts.

Check your submission to ensure everything is there, and that it is downloadable and openable.

Question	Topic	Points	Grade	Graded By
1	OO Analysis	15		
2	UML Class Diagram	15		
3	Software Architecture	10		
4	Design Pattern	10		
5	Design Principles	10		
6	UML Sequence Diagram	15		
7	Design Principles	10		
8	Design Principles	10		
9	Testing	10		
	TOTAL	**100**		

TEAM NAME/NUMBER: ______73____________

The following team members were present and participated in the final exam:

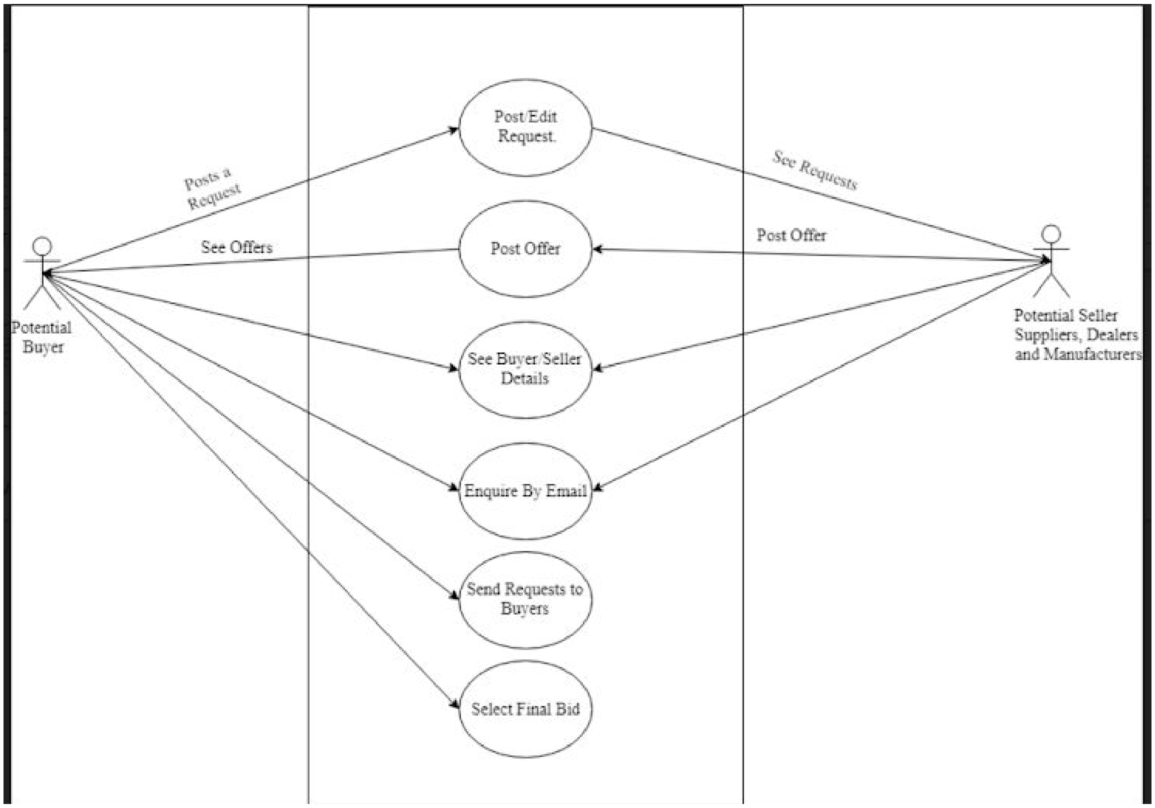
Printed Name	Signature
Zachary Waters	Zachary Waters
Kenedy Thorne	Kenedy Thorne
Chawalit Saetiew	Chawalit Saetiew
Collin Avidano	Collin Avidano
Conner Mathis	Conner Mathis

1. (15 Points) Perform a basic OO analysis of the reverse auction project. You should produce the following 3 artifacts:

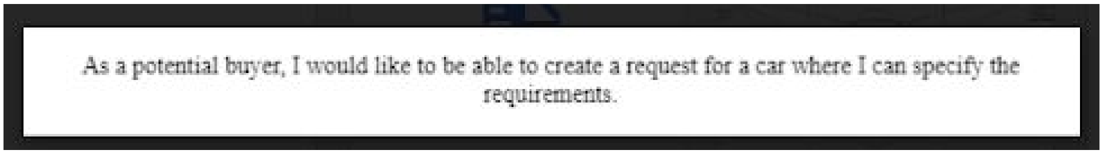
   a. A system context diagram identifying primary and supporting external actors that will interact with the system

   b. A complete user story of one scenario including the acceptance criteria.

   c. A domain model with stereotypes

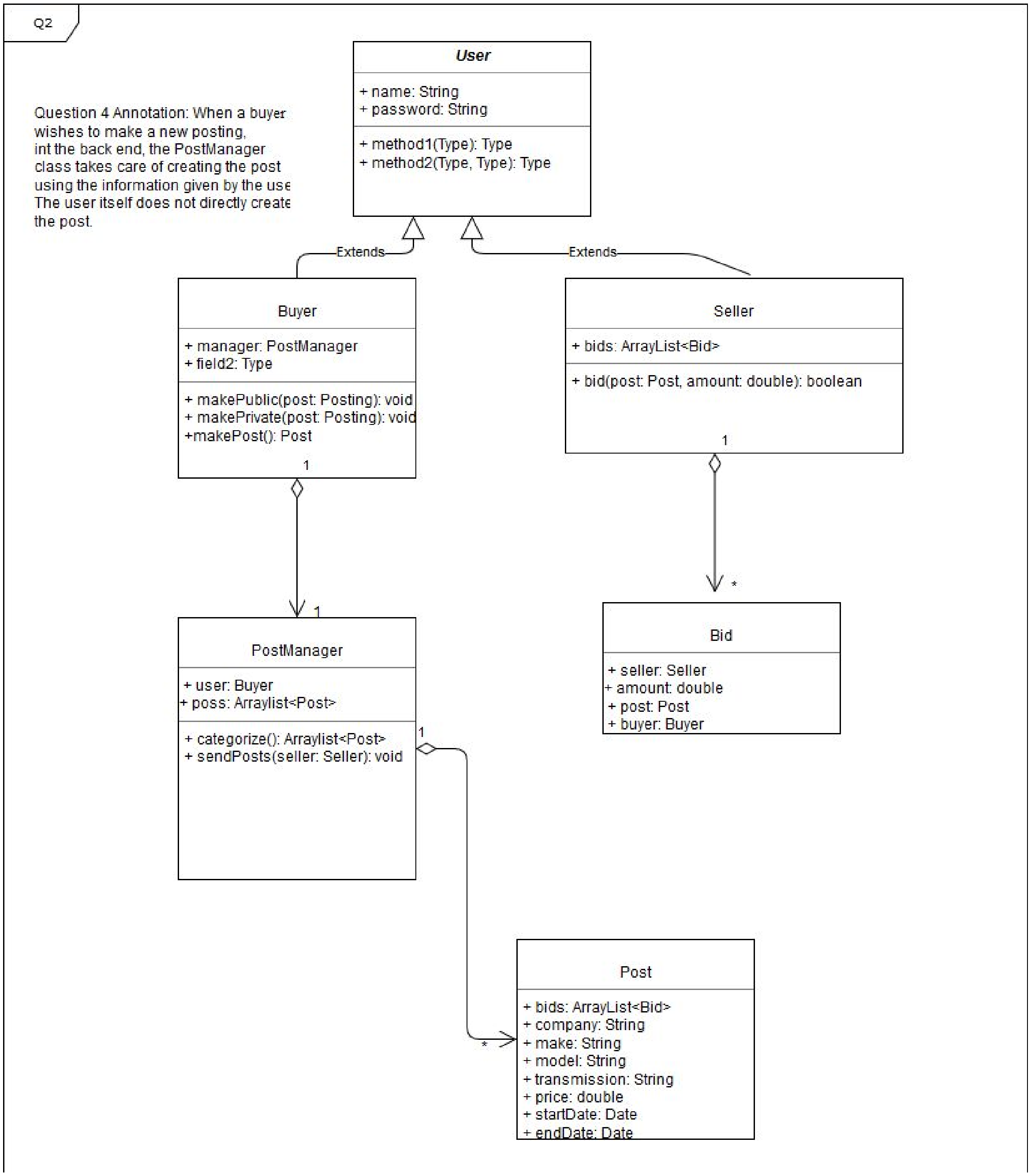


User Story:

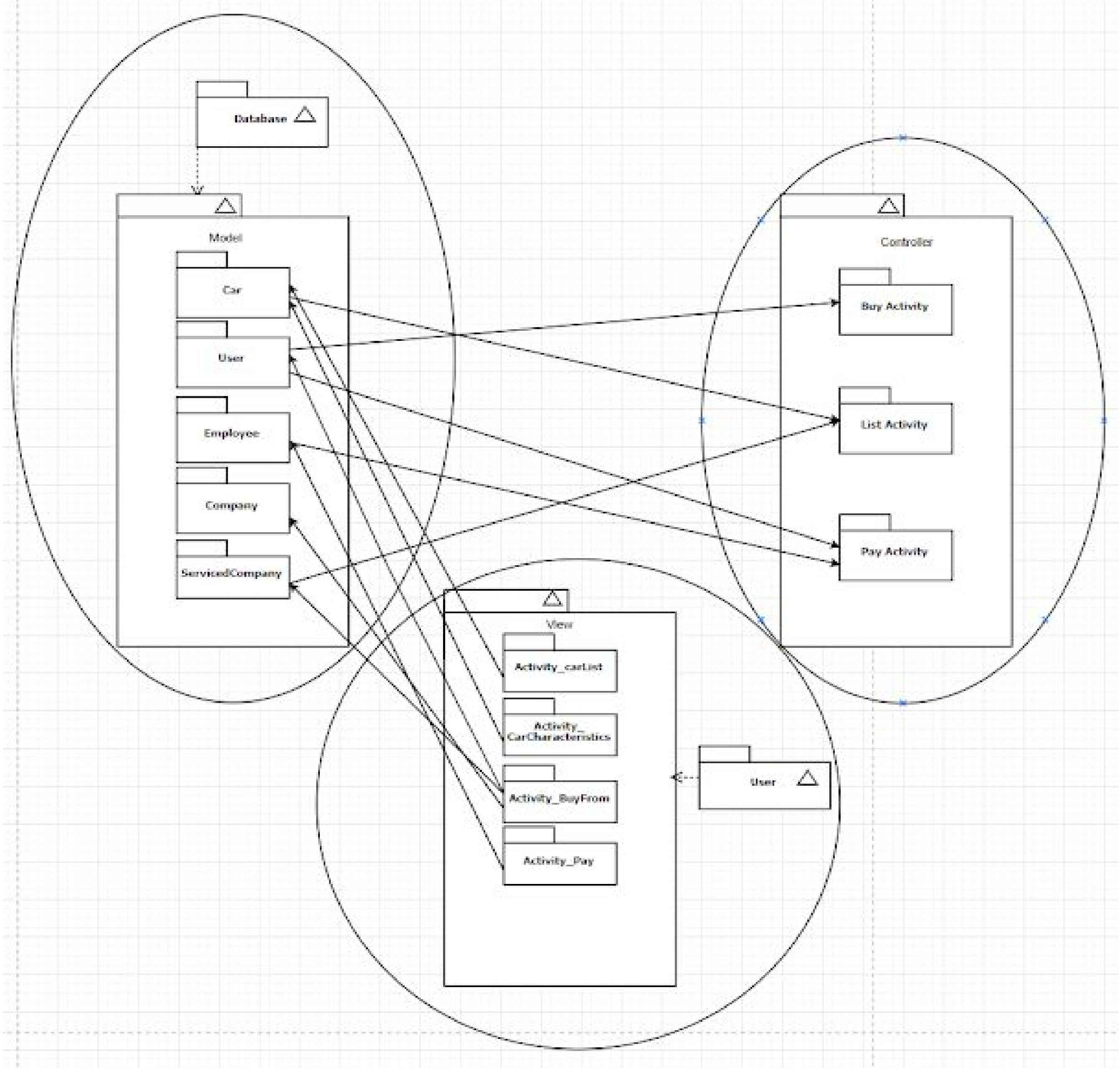


As a potential buyer, I would like to be able to create a request for a car where I can specify the requirements.

2. (15 Points) Create a UML class diagram for the reverse auction project. You should model the basic (back-end) classes in the application and ensure that your diagram has the following constructs: Inheritance, an Interface, an abstract class, a class which implements your interface, Instance Variables (4), Instance Methods (4) at least one of which takes parameters, one which has no parameters, one that returns something and one which returns nothing. (a method may satisfy more than one of these if desired). You should also show appropriate cardinality (multiplicity) constraints and associations. You do NOT need to model any networking/communication and UI classes. If you need such classes for a later question, you may abstract the entire UI or network interface into a single class.

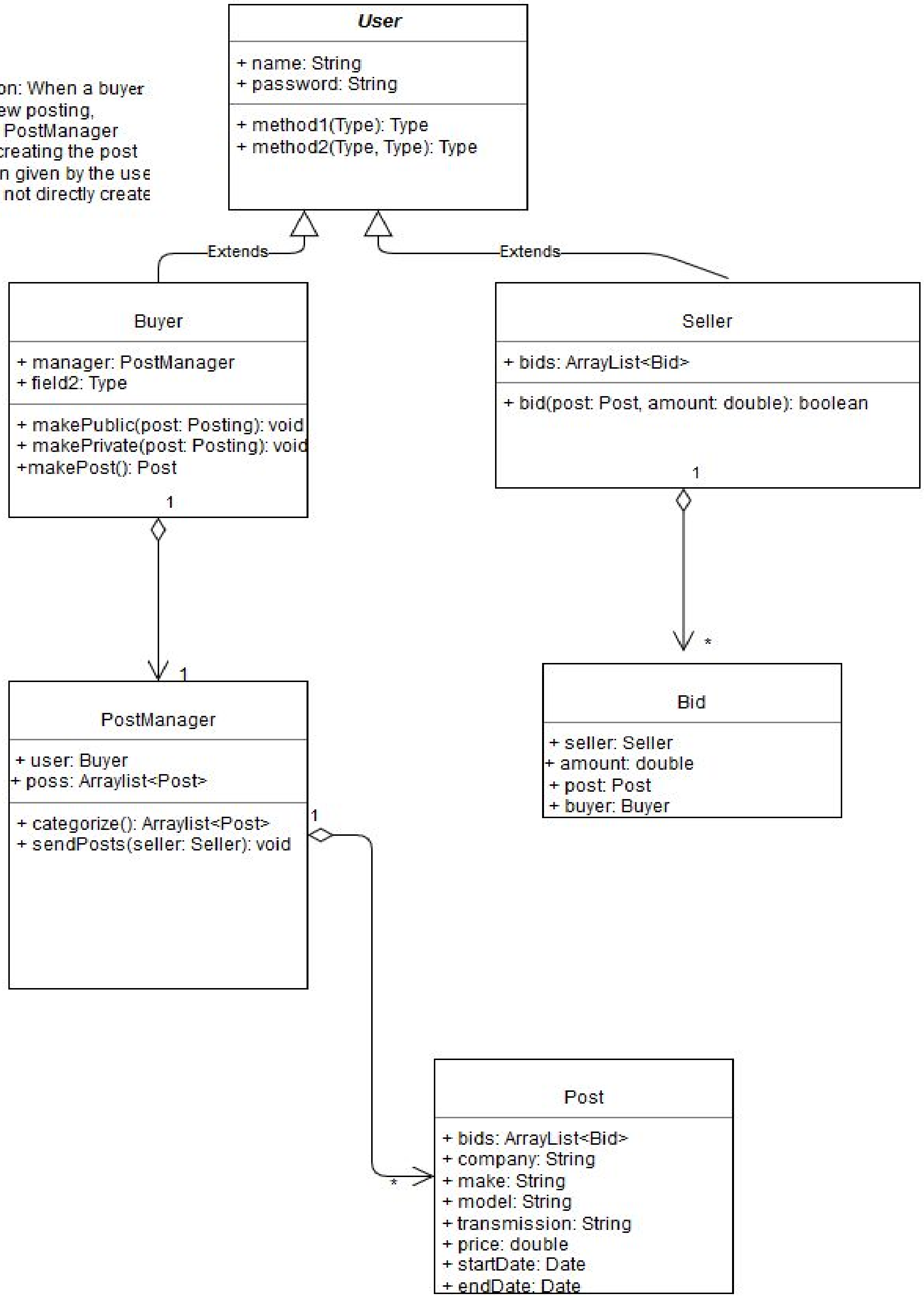

Q2
Question 4 Annotation: When a buyer wishes to make a new posting, int the back end, the PostManager class takes care of creating the post using the information given by the use The user itself does not directly create the post.
User
+ name: String
+ password: String
+ method1(Type): Type
+ method2(Type, Type): Type
Extends
Extends
Buyer
+ manager: PostManager
+ field2: Type
+ makePublic(post: Posting): void
+ makePrivate(post: Posting): void
+makePost(): Post
1
Seller
+ bids: ArrayList<Bid>
+ bid(post: Post, amount: double): boolean
1
*
1
PostManager
+ user: Buyer
+ poss: Arraylist<Post>
+ categorize(): Arraylist<Post>
+ sendPosts(seller: Seller): void
1
Bid
+ seller: Seller
+ amount: double
+ post: Post
+ buyer: Buyer
Post
+ bids: ArrayList<Bid>
+ company: String
+ make: String
+ model: String
+ transmission: String
+ price: double
+ startDate: Date
+ endDate: Date
*

3. (10 Points) Your project team has analyzed the application and determined that the main application should follow an architectural style. Discuss which style you would choose and how you would implement this style for your application by providing a package diagram.



4. (10 Points) Look at your design and UML diagram. Describe/show how you could apply the Command pattern to your design. Annotate the command pattern components on your UML diagram – this can be the same diagram as question 2 or you can make a special diagram just for this question.

Question 4 Annotation: When a buyer wishes to make a new posting, int the back end, the PostManager class takes care of creating the post using the information given by the use The user itself does not directly create the post.



5. (10 Points) Consider the open-closed and protected variation principles. Describe a variation point in your design and how you would address that requirement.

The database used can be one point of variation in our design. Because we need to store the posts and information about the users, owners, and manufacturers, a database would be perfect for our project. There are many different databases that could be used, so this is a variation point. Firebase, MongoDB, and an SQL database are a few of the options. Depending on how we want to use it, it could be possible we want to go from using one to the other.

6. (10 Points) For the user story you developed in question 1, draw a UML sequence diagram. Be sure the diagram is consistent with the UML class diagram and the user story you have previously completed.

7. (10 Points) The database for this application has been contracted out to Elbonia (nods to Scott Adams). The Elbonians are notorious for late delivery of shoddy software. You have to design the system and be able to test it potentially without having the actual database. Discuss how using a Facade or Adapter pattern and dependency injection would allow you to proceed on this project.

8. (10 Points) There are three code files provided on Canvas. Identify at least three violations of GRASP, SOLID, TDA or LOD. Correct the code and write a JUnit test for the method **computePartialPayroll** to ensure it has branch coverage.

9. (10 Points) Most student UI's for the semester homeless project allowed the users to type in the Lat/Long coordinates. Using Equivalence Partitions, describe the tests necessary to ensure that the data validation for this field is working.

Tests - latitude	Longitude
-91	0
91	0
-90	0
90	0
0	-181
0	-180
0	180
0	181

Extra Credit (3 points). Describe a good Android project that would be appropriate for 2340 next Fall. What would be in each of the 5 actual feature slices (M4, M5, M6, M7, M8)?