Lab 8: CS 370 Software Engineering

Team#:3

Team Project Title: GetGainz

	First Name	Last Name	CSUSM account ID	Contribution Percentage
1	Noya	Hafiz	201170234	25%
2	Carlos	Avila	200257842	25%
3	Nicholas	Brodsky	200324415	25%
4	Cherishma	Jalaparti	200827710	25%

Grading Rubrics (for instructor only):

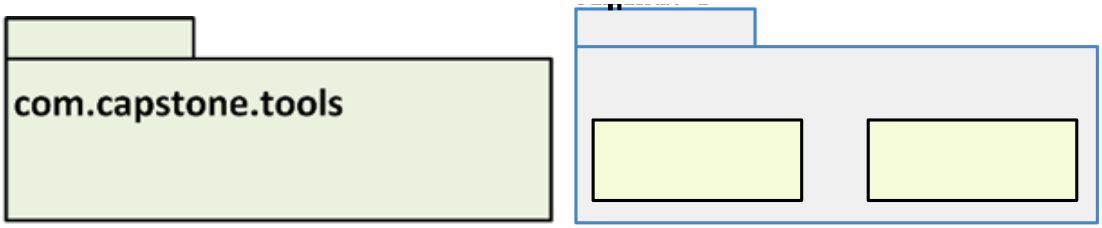
Criteria	1. Beginning	2. Developing	3. Proficient	4. Exemplary
Architecture Appropriateness	0-16	16-26	27-34	35-40
	Incorrect architectural decision	Need major changes	Need minor adjustment	The picked architecture style(s) is appropriate
Documentation notations	0-5	6-9	10-14	15-20
	Used wrong notations	Need major changes	Some minor issues	Used standard notations
Architecture design quality	0-16	16-26	27-34	35-40
	Missing important elements	Information provided is insufficient	Information provided is at appropriate mostly with Some minor issues	Information provided is at appropriate level of details
Total Grade (100)				

Problems:

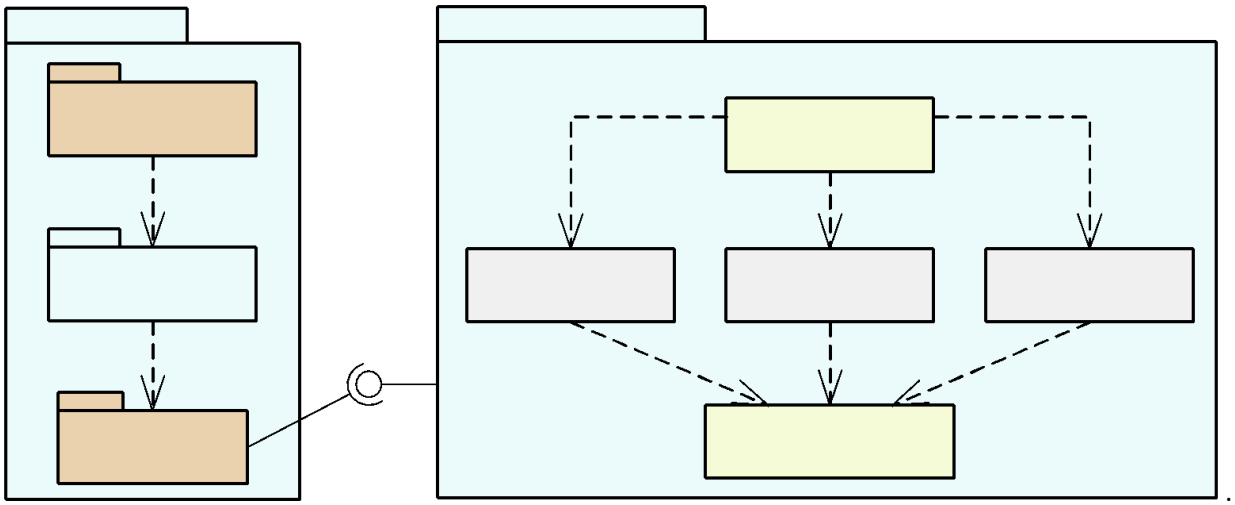
- **Project Progress monitor: Requirements (50%), Code (10%).**
- We have learned a few software architectures. Your team should work together to discuss which architecture might be appropriate for your project.

Lab 8: CS 370 Software Engineering

- We have learned three different documentation styles for documenting software system architecture. **A level-2 design in UML package diagram should expose the next level design elements pertinent to the top-level package. For example, in the figures below, the design on the left only shows the top-level package, whereas the diagram on the right exposes level-2 design elements.**



An example of level-2 design is shown below, where the uses-style is used



Use a UML package diagram to document the module view of your level-2 design of system architecture.

Instructions:

- **Use the Lucid tool** as part of your Canvas account **to create the solution and export it as PDF**.

Lab 8: CS 370 Software Engineering

- Please **upload** your solution in PDF to this graded assignment.
- **One submission from each team**.

We have used the client-server architectural pattern.