*Andreas: 1 + 3, Ian: 4 + 5, Pontus: 6 + 7, Jessie: 8 + 9, Tommy: 10 + 11, Shan: 12 + 13*

**3. For each of the 12 assignment reports you have at hand, analyze the proposed architectural solution with respect to:**

**a. Risks -** Having the system in one place makes it vulnerable to power outage**.**

**b. Non-risks -** Security using client-server (2)

**c. Sensitivity points -** Credibility is achieved by using an external payment provider

**d. Tradeoff points -** Lower cost vs availability, having no backup

ATAM: Outputs

**Risk**

Architectural decision that may lead to undesirable

consequences

**Non risk**

Architectural decision that is deemed safe

A general concern of a group of interrelated risks in a

design, assigned its own risk value

**Sensitivity point**

A property of one or more components (and/or

component relationships) that is critical for achieving a

particular quality attribute response

**Tradeoff point**

An architectural decision that affects more than one

quality attribute (possibly in opposite ways)

**4. Review each of the 12 proposed solutions with respect to the following abstract design principles:**

**a. Information hiding** (people need not see) **Yes**

**b. Minimize coupling** (components don’t link everywhere and don’t have unnecessary connections) **Yes**

**c. Coherence** (components that work the same thing are together) **Yes**

**d. Divide and conquer** (System parts/components are broken down into manageable parts) **Yes**

**e. Separation of concerns** (components functions are in the component and not spread around the system) **yes**

**f. Keep it simple Yes**

**g. No circular dependencies** (Components are not connected in a way that two or more all depend on eachother) **Yes**

**h. Layering** (separating front, back and db) **Yes**

**i. Modularity** (parts can be used in another project or be switched out)

**Yes**

**5. Give an overall grade for each architectural solution at the scale of 1 (poor) to 10 (excellent).**

**9**

**Justify your score in terms of your answers to the previous questions**.

The program has addresses separation of concerns an embodiment of layering. The program is modular and together provides encapsulation and which means there is information hiding.

Notes:

MoSCoW Method:

– M - MUST: Describes a requirement that must be satisfied in the

final solution for the solution to be considered a success.

– S - SHOULD: Represents a high-priority item that should be

included in the solution if it is possible. This is often a critical

requirement but one which can be satisfied in other ways if strictly

necessary.

– C - COULD: Describes a requirement which is considered desirable

but not necessary. This will be included if time and resources

permit.

– W - WON'T: Represents a requirement that stakeholders have

agreed will not be implemented in a given release, but may be

considered for the future.

**Architectural Styles**

Allocation:

– Map-Reduce

– Multi-tier

• Module:

– Layered

• C&C:

– Broker

– Model-View-Controller

– Pipe-and-Filter

– Client-Server

– Peer-to-Peer

– Service-Oriented Architecture

– Publish-Subscribe

– Shared-Data

– Blackboard