yazzlkja

Photographer Matching Platform

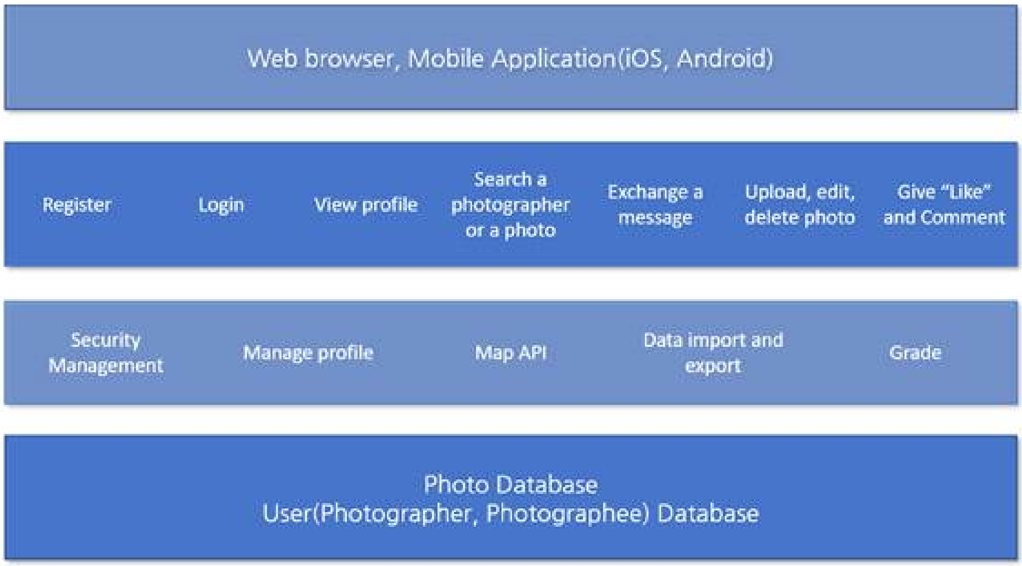
Architecture

Team 2

2014210011	고나현	rhskgus@korea.ac.kr
2013210086	김영훈	2013210086@korea.ac.kr
2014210101	김원경	kwk2392@korea.ac.kr
2013210081	박진호	onlylove6913@kroea.ac.kr
2013210039	이민섭	alexn0@korea.ac.kr
2014210065	이상진	eesj@korea.ac.kr
2015410005	이정섭	james319@korea.ac.kr
2013210020	임원준	rhimjun@korea.ac.kr
2014210059	정희석	poco2889@korea.ac.kr

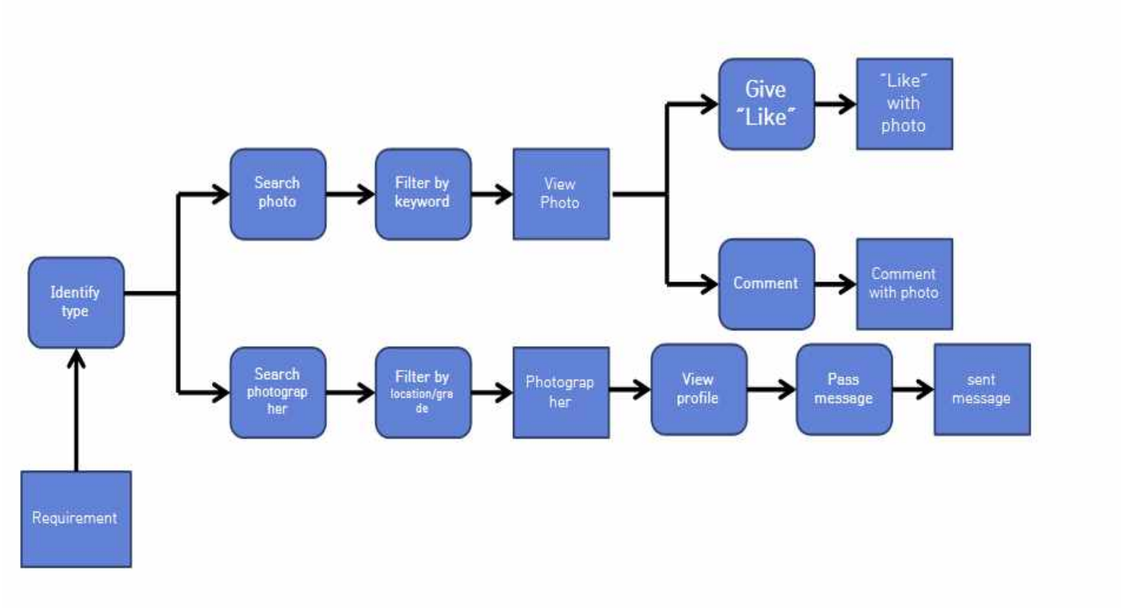
Table of Contents

Ⅰ. Layered Architecture



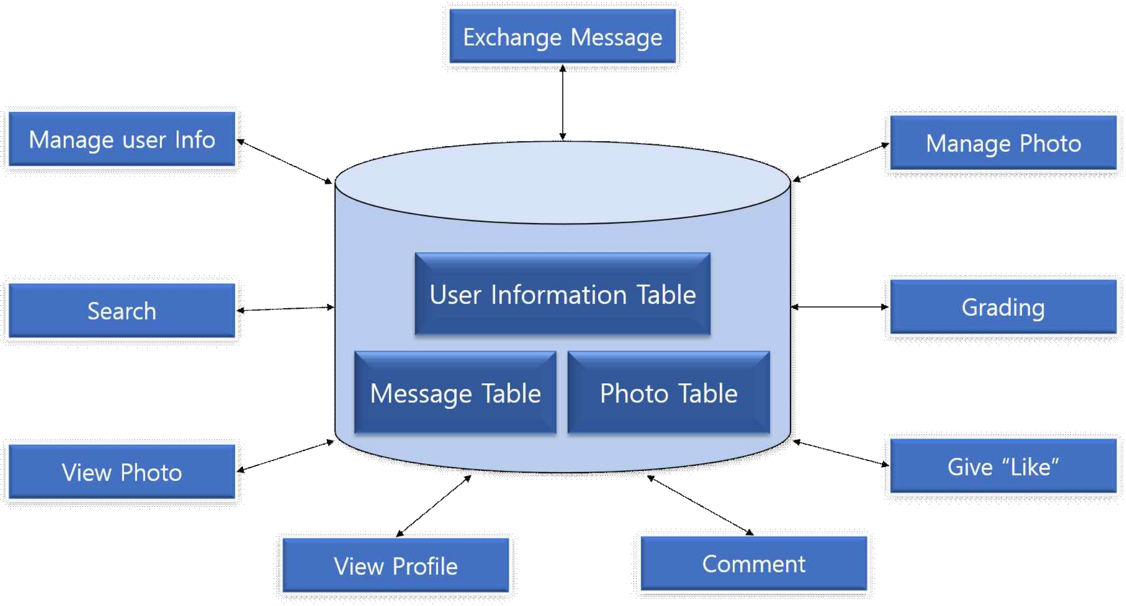
Description	Organizes the system into layers with related functionality associated with each layer. A layer provides services to the layer above it so the lowest-level layers represent core services that are likely to be used throughout the system.
Advantages	Easy to understand that layer communicates with adjacent layer. For instance, we can intuitively find out that 'Login' which is on user interface management layer(2nd layer) is related to 'Security Management' which is on functionality layer.
Disadvantages	It is confusing to find the right levels of abstraction. For example, 'Exchange a message', which is on 2nd layer, includes some functionality factors. As it is normally on 3rd layer, 'Exchange a message' should be placed on 3rd layer. But messaging is too ambiguous to put 3rd layer because its properties are also into 2nd layer.

Ⅱ. Pipe and Filter Architecture



Description	The processing of the data in a system is organized so that each processing component (filter) is discrete and carries out one type of data transformation. THe data flows (as in a pipe) from one component to another for processing
Advantages	It is comprehensive and understandable. It makes users easy to recognize how data flow in the system. Easy to understand and supports transformation reuse. Workflow style matches the structure of many business processes. Evolution by adding transformations is straightforward. Can be implemented as either a sequential or concurrent system.
Disadvantages	Some functions like “Pass Message” which requires interactive inputs are difficult to represent with Pipe and Filter pattern. It applies same for “Manage profile”, “Grade”. The format for data transfer has to be agreed upon between communicating transformations. Each transformation must parse its input and unparse its output to the agreed form. This increases system overhead and may mean that it is impossible to reuse functional transformations that use incompatible data structures.

Ⅲ. Repository Architecture



Description	TAll data (user information, messages, photos) in a system is managed in a central repository that is accessible to all system components. Components do not interact directly, only through the repository. Sub-systems must exchange data. This may be done in two ways : Shared data is held in a central database or repository and may be accessed by all sub-systems. : Each sub-system maintains its own database and passes data explicitly to other sub-systems. When large amount of data are to be shared, the repository model of sharing is most commonly used a this is an efficient data sharing mechanism.
Advantages	Components can be independent – they do not need to know of the existence of other components. Changes made by one component can be propagated to all components. All data can be managed consistently (e.g., backups done at the same time) as it is all in one place.
Disadvantages	The repository is a single point of failure so problems in the repository affect the whole system. May be inefficiencies in organizing all communication through the repository. Distributing the repository across several computers may be difficult.

Ⅳ. Our Architecture - Repository Architecture

In case of Layered Architecture, It is confusing to find the right levels of abstraction. For example, 'Exchange a message', which is on 2nd layer, includes some functionality factors. As it is normally on 3rd layer, 'Exchange a message' should be placed on 3rd layer. But messaging is too ambiguous to put 3rd layer because its properties are also into 2nd layer.

And in case of Pipe and Filter Architecture, Some functions like "Pass Message" which require interactive inputs are difficult to represent with Pipe and Filter pattern. It applies same for "Manage profile", "Grade".

But repository Architecture can overcome the above disadvantages and additionally has the following advantages. Components can be independent – they do not need to know of the existence of other components. Changes made by one component can be propagated to all components. All data can be managed consistently (e.g., backups done at the same time) as it is all in one place. Consequently, repository architecture is appropriate for our team project.