Team Project – Team3

**Architectural Design**

**of Bekoor6**

Dec. 12, 2019

**Team Members**

|  |  |
| --- | --- |
| **Name** | **Email** |
| Kou Yanran | ehdals338@gmail.com |
| Youngkyoung Kim | Agnes66k@naver.com |
| Juhyung Kim | kjhkjh75@naver.com |

Index

[Team Project – Team3 1](#_Toc27001399)

[1. Introduction 4](#_Toc27001400)

[1.1. Needs 4](#_Toc27001401)

[1.2. Bekoor6 5](#_Toc27001402)

[2. System Architecture 5](#_Toc27001403)

[2.1. Overall Architecture of Bekoor6 5](#_Toc27001404)

[2.2. Context of System 6](#_Toc27001405)

[3. Architectural drivers 8](#_Toc27001406)

[3.1. Design Purpose 8](#_Toc27001407)

[3.2. Use Case Model 8](#_Toc27001408)

[3.3. Quality Attributes 10](#_Toc27001409)

[3.4. Architectural Concerns 10](#_Toc27001410)

[3.5. Constraints 11](#_Toc27001411)

[4. Review Inputs 11](#_Toc27001412)

[5.1. Establish iteration goal by selecting drivers 12](#_Toc27001413)

[5.2. Choose one or more elements of the system to refine 12](#_Toc27001414)

[5.3. Choose one or more design concepts design concepts that satisfy the selected drivers 13](#_Toc27001415)

[5.4. Instantiate architectural elements, allocate responsibilities, and define interfaces 13](#_Toc27001416)

[5.5. Sketch views and record design decisions 14](#_Toc27001417)

[5.6. Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose 16](#_Toc27001418)

[5. ADD Iteration 2 – Identifying structures to support primary functions 17](#_Toc27001419)

[6.1. Establish iteration goal by selecting drivers 17](#_Toc27001420)

[6.2. Choose one or more elements of the system to refine 17](#_Toc27001421)

[6.3. Choose one or more design concepts design concepts that satisfy the selected drivers 17](#_Toc27001422)

[6.4. Instantiate architectural elements, allocate responsibilities, and define interfaces 18](#_Toc27001423)

[6.5. Sketch views and record design decisions 18](#_Toc27001424)

[6.6 Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose 21](#_Toc27001425)

[7. Iteration 3 – Addressing Quality Attribute Scenario Driver (QA-5) 21](#_Toc27001426)

[7.1 Establish Iteration Goal by Selecting Drivers 21](#_Toc27001427)

[Based on the fundamental structural decisions made in iterations 1 and 2, we can now start to reason about the fulfillment of some of the more important quality attributes 21](#_Toc27001428)

[7.2 Choose One or More Elements of the System to Refine 21](#_Toc27001429)

[For this iteration, the architect focuses on the QA-5 quality attribute scenario: 21](#_Toc27001430)

[- streaming delay should be less than 5 seconds. 21](#_Toc27001431)

[7.3 Choose One or More Design Concepts that Satisfy the Selected Drivers 21](#_Toc27001432)

[The elements that will be refined are the physical nodes that were identified during the first iteration: 21](#_Toc27001433)

[- Streaming Server / Application Server / Web Server 21](#_Toc27001434)

[- Streaming database 21](#_Toc27001435)

[7.4 Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces 22](#_Toc27001436)

[7.5 Sketch Views and Record Design Decisions 23](#_Toc27001437)

[물어보기 23](#_Toc27001438)

[Sequential diagram 24](#_Toc27001439)

[7.5 Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose 25](#_Toc27001440)

[\*Definitions 26](#_Toc27001441)

[8. Iteration 4 – Addressing Quality Attribute Scenario Driver (QA-1) 28](#_Toc27001442)

[8.1 Establish Iteration Goal by Selecting Drivers 28](#_Toc27001443)

[8.2 Choose One or More Elements of the System to Refine 28](#_Toc27001444)

[8.3 Choose One or More Design Concepts that Satisfy the Selected Drivers 28](#_Toc27001445)

[8.4 Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces 29](#_Toc27001446)

[8.5 Sketch Views and Record Design Decisions 30](#_Toc27001447)

[8.6 Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose 31](#_Toc27001448)

[9. Iteration 5 – Addressing Quality Attribute Scenario Driver (QA-8) 33](#_Toc27001449)

[9.1 Establish Iteration Goal by Selecting Drivers 33](#_Toc27001450)

[9.1 Choose One or More Elements of the System to Refine 33](#_Toc27001451)

[9.2 Choose One or More Design Concepts that Satisfy the Selected Drivers 33](#_Toc27001452)

[9.3 Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces 34](#_Toc27001453)

[9.4 Sketch Views and Record Design Decisions 35](#_Toc27001454)

[Below is payment sequence diagram. The payment gateway performs control and monitoring of the payment process as a whole and responds to the occurrence of an error situation during the transaction. 35](#_Toc27001455)

[9.5 Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose 36](#_Toc27001456)

[10. Iteration 6 – Addressing Quality Attribute Scenario Driver (QA-3) 37](#_Toc27001457)

[10.1 Establish Iteration Goal by Selecting Drivers 37](#_Toc27001458)

[Business goal of our system is to stimulate users' desire to purchase using live streaming. If user can access to our system using various devices (PC and mobile), we can get more chance to make profit. In other word, we should support PC website and mobile application (or website) both. At the same time, we should not change original architecture too much while add some modules that support portability. Therefore, our goal in iteration 6 is modifying the architecture to support usability (QA-3) and support as many devices as possible which is of Android operating system (CON-3). 37](#_Toc27001459)

[10.2 Choose One or More Elements of the System to Refine 37](#_Toc27001460)

[In iteration 6, quality attribute scenario 3 (QA-3: portability) will be discussed. 37](#_Toc27001461)

[1. Bekoor 6 should be produced for different computing platforms, such as Mobile APP, Mobile web browser and Desktop web browser. 37](#_Toc27001462)

[2. Also, Bekoor 6 should be able to move across different environment, not just across platforms. For example, modify software and make it adaptable to work on Linux, Windows, Mac OS, Unix, etc. 37](#_Toc27001463)

[Hence, the system interfaces and application logic will be analyzed respectively. 37](#_Toc27001464)

[10.3 Choose One or More Design Concepts that Satisfy the Selected Drivers 37](#_Toc27001465)

[10.4 Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces 39](#_Toc27001466)

[10.5 Sketch Views and Record Design Decisions 39](#_Toc27001467)

[10.6 Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose 41](#_Toc27001468)

[10.7 41](#_Toc27001469)

[11. Iteration 7 41](#_Toc27001470)

[11.1 Establish Iteration Goal by Selecting Drivers 41](#_Toc27001471)

[11.2Choose One or More Design Concepts that Satisfy the Selected Drivers 42](#_Toc27001472)

[11.3 Choose One or More Design Concepts that Satisfy the Selected Drivers 42](#_Toc27001473)

[11.4 Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces 42](#_Toc27001474)

[11.5 Sketch Views and Record Design Decisions 43](#_Toc27001475)

[43](#_Toc27001476)

[11.6 Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose 43](#_Toc27001477)

1. Introduction

This section briefly describes the target system including a description of the technical terms to be used in the future.

* 1. Needs

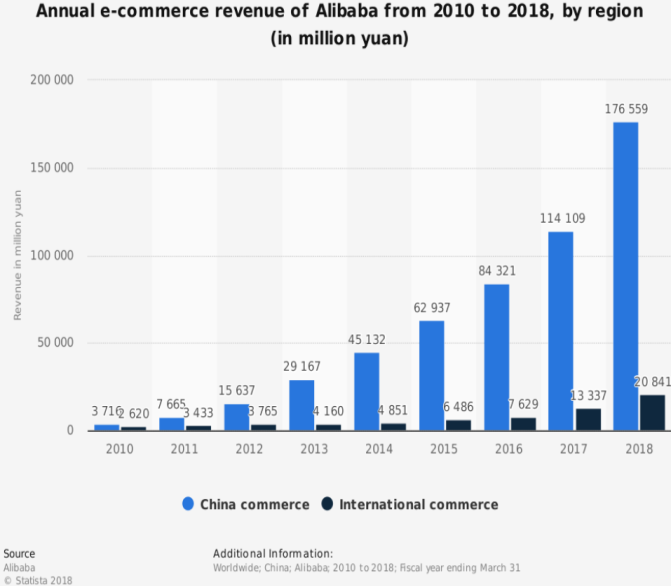
Nowadays, online shopping mall is essential for apparel companies. Due to the advance of internet and mobile phone, users prefer to do shopping with mobile application. Mobile shopping mall application provide functions for users convenient shopping. Customers can check details of the product at their home. Also they can pay online so they don’t have to go to the store. This enables users to shop easily and everywhere. Hence more and more users are using mobile application for shopping. As shown in figure2, 259% of users shifted to mobile shopping than using desktop.

Figure 1. Annual e-commerce revenue of Alibaba from 2010 to 2018(in million yuan)

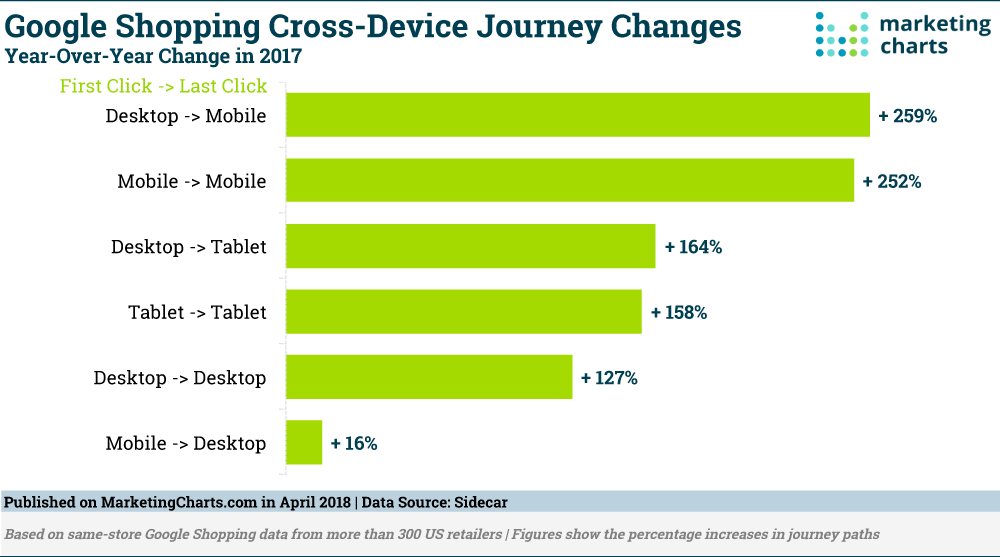


Figure 2.

However there is a need for detailed specification of their product. Due to demand, recent systems apply ‘live streaming’ function to their application. The trend dominates the market in China. In Taobao, the world’s biggest e-commerce website, 81 top live commerce generated more than $15 million in sales. In South Korea, the LF Corp launched live streaming shopping with real0time chatting and one-click purchase. This has resulted in a 30% year-over-year increase since 2015.

* 1. Bekoor6

Bekoor6 is a shopping and live commerce system. In their live video streaming, buyers can interact with and order from sellers in real time. The advantage of live streaming is that they can introduce their products real-time, and give feedback to customers directly. So users may have an immediacy to get the desired product information instantly. The buyer can solve the question of the product by asking the seller about the durability test and the wearing of the clothes. Even after the live streaming is finished, even if the buyer does not watch live while uploading the live video to the seller's store, they can check the product information while watching the uploaded video. The managers can choose the group of the customers for product promotion.

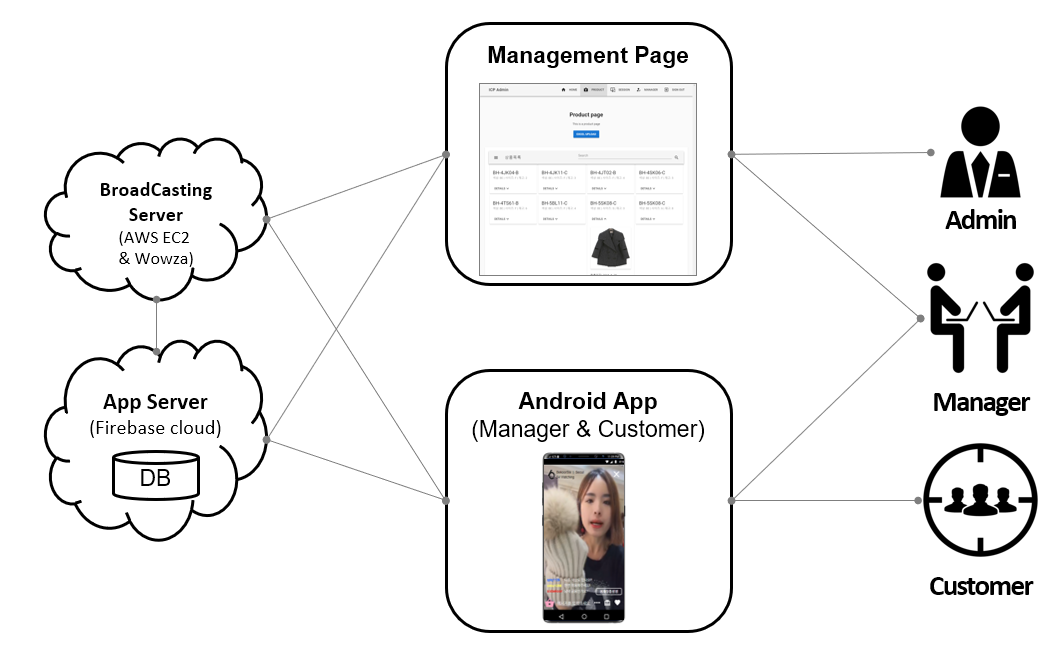
1. System Architecture

This section describes architecture of entire system.

* 1. Overall Architecture of Bekoor6

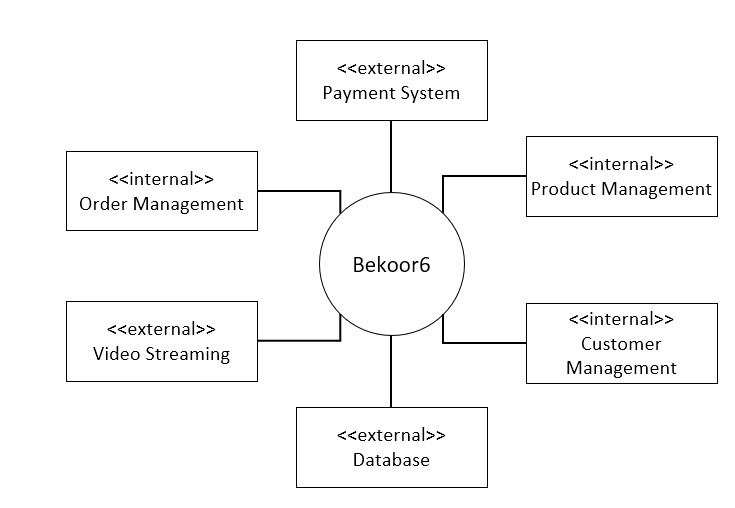
*Bkoor6 have* two types of server which are ‘Broadcasting Server’ and ‘App Server’. The App Server provides typical functions of online-shop. It manages information of products and payment transactions.

The ‘Broadcasting Server’ is only dedicated for video streams. It is physically separated from the App Server because it needs to handles large sized multi-media. The data can be temporarily cached to proxy servers for providing better user experience.

Figure 3. System Overview

* 1. Context of System

Internal and external systems which communicate with *Bekoor6* is described in the context diagram and table below.



|  |  |
| --- | --- |
| System | Description |
| Payment System | An external system which handles payment transactions |
| Database | An external system which stores every kind of data |
| Video Streaming | An internal system dedicated for managing video streaming service |
| Product Management | An internal system which manages products selling in *Bekoor6* |
| Order Management | An internal system which manages orders and stores history |
| Customer Management | An internal system which manages customers in *Bekoor6* |

1. Architectural drivers
   1. Design Purpose

This system is greenfield system in mature domain. The purpose is to yield a detailed design to support the construction of the system.

Business goal is to promote and sell products to customers through live streaming and video. Customers should be able to purchase the preferred product in video easily. Manager should be able to promote their product effectively by live streaming to purchasable customer group. We should combine e-commerce system and live streaming

* 1. Use Case Model

This section contains the use case modeling for the Bekoor6. Important use case scenarios based on the actors have been included here. This use case model depicts relationship between the actors and their role in Bekoor6. Not only live streaming functionalities also order and payment functionalities are included in use case.

**텍스트이(가) 표시된 사진

자동 생성된 설명**

|  |  |
| --- | --- |
| **Use Case** | **Description** |
| 1 Open Broadcasting | Manager open a Broadcasting channel |
| 2 Set Favorite | Customers can add the channel in favorite menu then you can get several notification |
| 3 Browse Broadcasting | Customer browse the list of live streaming |
| 4 Enter Broadcasting | Customer enter into a Broadcasting. |
| 5 Live chatting | Customer or manger can write the message and show message to other`s real time. |
| 6 Start Live Streaming | Manager starts live streaming broadcasting |
| 7 Stop Live Streaming | Manager stops live streaming broadcasting |
| 8 Share Broadcasting | Customer share streaming through SNS |
| 9 Click like Button | Customer click like button while watching streaming. |
| 10 View Number of Viewers | Manager and customer who watches live sees how many users are watching the video now. |
| 11 Browse Items | Customer browses product items in the application. |
| 12 Upload New Item | Manager add new items |
| 13 Add Item to Cart | Customer add an item to cart |
| 14 Add Item to Wishlist | Customer add an item to wish-list |
| 15 Buy Item Now | Customer buys items in the cart. |
| 16 Checkout Payment | Payment systems check out payment with provided payment information. |
| 17 Share the items through SNS | Customer can share the information about items which he or she like to their friends through SNS like kakao,facebook |
| 18 Notice | Admin upload notice for all customers. |

* 1. Quality Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Quality Attribute** | **Scenario** | **associated UC ID** |
| QA-1 | Security | A user of the system performs any action in the system such as updating personal information, banking information or any purchase activity should be visible only to them. | UC 16 |
| QA-2 | Performance | when a user enters a key word in input box and search product, the matched product list should be shown with an acceptable time range | UC 11 |
| QA-3 | Usability | A user should be able to access Bekoor 6 from different devices like mobile, desktop and web. | All use cases |
| QA-4 | performance | audio and video should be synchronized while streaming playing | UC 4, 6, 7 |
| QA-5 | performance | streaming delay should be less than 5 seconds. | UC 4, 5 |
| QA-6 | availability | the system shall continue operating with more than maximum user number accessing the system at the same time. | UC 4, 10 |
| QA-7 | extensibility | the system shall support adding new data sources by just updating a configuration. with no interruption of ongoing data collection. | UC 12, 13, 14 |
| QA-8 | Safety | payment system is the most important function of the in our system. So, Keeping safety of the this system without failure also important in payment process and architectural design decision. In this iteration, we concern about the payment safety attributes | UC16 |
| QA-9 | Security | a user log in the web service. authentication function should be performed to make sure that this user is an admin or manager. | UC16,17 |
| QA-10 | Usability | provide proper view for management like dashboard view templates etc. | UC6,12 |

* 1. Architectural Concerns

|  |  |
| --- | --- |
| **ID** | **Concerns** |
| CRN-1 | Establishing an overall initial system architecture. |
| CRN-2 | Leverage the expertise of the development team's knowledge of Java and Java-related technologies. |
| CRN-3 | Allocate tasks based on the strengths of each member of the development team. |
| CRN-4 | Introduction of new functionality must, as much as possible, avoid modifications to the existing core functionality. |

* 1. Constraints

|  |  |
| --- | --- |
| **ID** | **Constraints** |
| CON-1 | The web application must be compatible to Internet Explorer and Chrome |
| CON-2 | The system should be able to work on at least Samsung and LG |
| CON-3 | Most of the modules should be unit tested |

2. Review Inputs

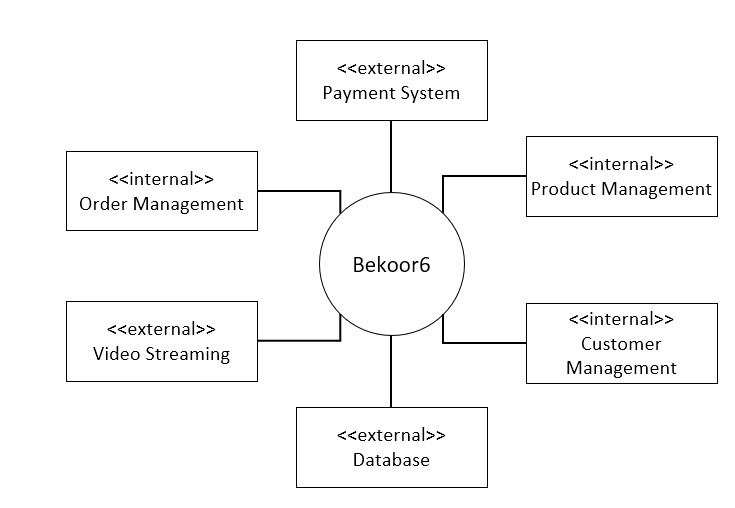
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Quality attribute scenarios | |  |  |  | | --- | --- | --- | | **Scenario ID** | **Importance to the customer** | **Difficulty of implementation according to the architect** | | QA-1 | High | Medium | | QA-2 | High | Low | | QA-3 | High | High | | QA-6 | Medium | Medium | | QA-10 | Medium | Low | |
| Constraints | All of constraints mentioned above all chosen as architectural driver. |
| Architectural Concerns | All architectural concerns discussed in above contents are selected as architectural drivers. |

1. ADD Iteration 1 – Establishing overall system structure
   1. Establish iteration goal by selecting drivers

|  |  |
| --- | --- |
| **Goal** | Derive overall structure of the system considering all of the drivers that might influence the structure of the system. |
| **Selected Drivers** | QA-1: Security / QA-2: Performance / QA-6: Availability / QA-8: Safety  CON-1: Web browser compatibility  CON-2: Mobile phone brand compatibility  CRN-1: Establishing an overall initial system architecture |

* 1. Choose one or more elements of the system to refine

This is a greenfield system as live e-commerce system is well-known these days. So the items to be refined are all of internal systems.



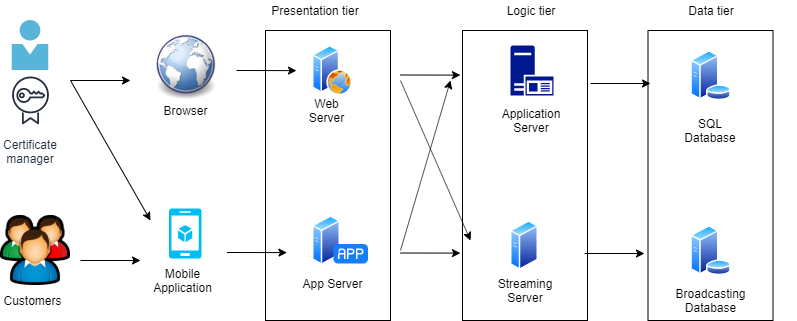
* 1. Choose one or more design concepts design concepts that satisfy the selected drivers

|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale and assumptions** |
| Using “N-tier architecture” as reference architecture. | “N-tier architecture” may include the web server that delivers web pages to the consumer’s browser, application servers(UC-1), database servers(UC-13), and any other underlying servers or devices. We can also adopt video streaming service(UC-6) and add more components without having to rewrite the entire application or redesigning whole software.  Meanwhile, in terms of security, sensitive or confidential information in the logic tier can be stored, keeping it away from the presentation tier, thus making it more secure. (QA-1) |

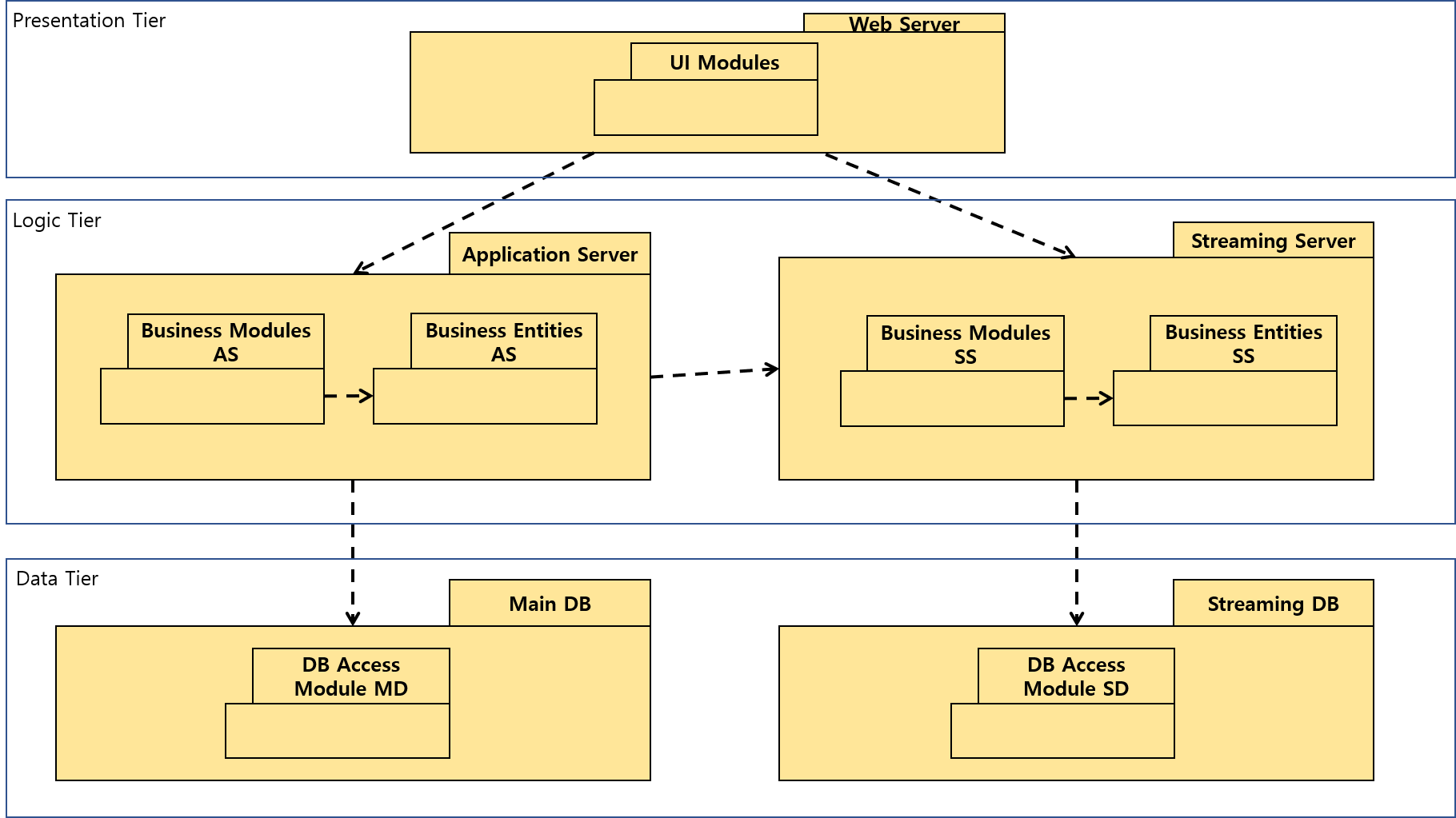
* 1. Instantiate architectural elements, allocate responsibilities, and define interfaces

|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale** |
| Instantiate a presentation layer (web) ,  a logic layer (application), a data-storage layer. | We need typical 3 layers for e-commerce system.  First, presentation layer is publicly accessible and is used to present information such as web pages, forms, advertisements, merchandise, and shopping cart contents to the consumer’s web browser. Because web servers are publicly accessible, sensitive or confidential information—such as payment card data—must never be stored on web servers.  Second, logic layer (application server) performs a variety of processing functions and should never be publicly accessible. In most cases, consumers do not interact directly with application servers, as the application servers receive requests from the web server, to process, format, and prepare data for storage or transmission. Application servers may also receive responses or retrieve content from database servers and subsequently pass the results back to the web server for presentation to the consumer.  The data-storage tier includes database servers and any other system or media used to store data. Since databases may store sensitive information, including payment card data, database servers must never be publicly accessible. In a three-tier computing model, the database only accepts requests from and provides responses to properly formatted and authenticated requests, usually made by an application server. |
| Create new streaming server for live video streaming. | The system is a live e-commerce system, so the system should include streaming server for live video streaming service (UC-6). Extend the module for streaming server and add new data storage to manage media data. |

* 1. Sketch views and record design decisions



|  |  |
| --- | --- |
| **Element** | **Responsibility** |
| Browser | It is responsible for the visual interface between the manager and the web server. (such as internet explorer, chrome, firefox…) |
| Mobile application | It is responsible for the visual interface between the customer and the application server` |
| Presentation Tier | This tier contains modules that control user interaction and use case control flow. |
| Logic Tier | This tier contains modules that perform business logic operations that can be executed locally |
| Data Tier | This tier contains modules that are responsible for communication with the server. |
| Web Server | It is accessible only to managers and are used to present information such as web pages, forms, merchandise, customer information, broadcast management to the manager’s web browser. |
| App Server | It is publicly accessible and are used to present information such as web pages, forms, advertisements, merchandise, and shopping cart contents to the consumer’s web browser. |
| Application Server | It performs a variety of processing functions and should never be publicly accessible. |
| SQL database | It should store information about product items, payment and personal information. |
| Stream server | It performs live video streaming services. |
| Broadcasting database | It should store media data(video) for video streaming service. |



|  |  |
| --- | --- |
| **Element** | **Responsibility** |
| UI Modules | These modules are for the user interface and receive user inputs. |
| Business Modules AS | These modules implement business operations. (User identity information, Payment and order management) |
| Business Entities AS | These entities make up the business model.(User account, Payment/Order information) |
| Business Modules SS | These modules implement streaming service. (Live video service, chatting service) |
| Business Entities SS | These entities implement streaming data.(Multimedia, interactive chat text) |
| DB Access Module MD | This module is responsible for people identity information, payment and order information. (UC-13,15) |
| DB Access Module SD | This module is responsible for storing video in the streaming service. |

* 1. Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

|  |  |  |  |
| --- | --- | --- | --- |
| **Not addressed** | **Partially addressed** | **Completed addressed** | **Design decisions Made During the Iteration** |
|  |  | CRN-1 | Overall initial architecture has been formed from the N-tier reference architecture. |
|  | UC-6 |  | Selected reference architecture establishes the modules that will support this functionality. |
|  | UC-1 |  | Selected reference architecture establishes the modules that will support this functionality. |
|  | UC-13 |  | Selected reference architecture establishes the modules that will support this functionality. |
|  | QA-1 |  | The elements that support the associated use case UC-16 have been identified. |
| QA-6 |  |  | No relevant decision made, as it is necessary to identify the elements that participate in the use case that is associated with the scenario. |

1. ADD Iteration 2 – Identifying structures to support primary functions
   1. Establish iteration goal by selecting drivers

|  |  |
| --- | --- |
| **Goal** | Identifying structures to support primary functionalities. |
| **Selected Drivers** | CRN-3: Allocate tasks based on the strengths of each member of the development team  Channel management related use-cases: UC-1  Video related use-cases: UC-3 ~ UC-7  Item management use-cases: UC-12 |

* 1. Choose one or more elements of the system to refine

The elements that will be refined in this iteration are the modules located in the different tiers defined by the architecture derived from previous iteration.

* 1. Choose one or more design concepts design concepts that satisfy the selected drivers

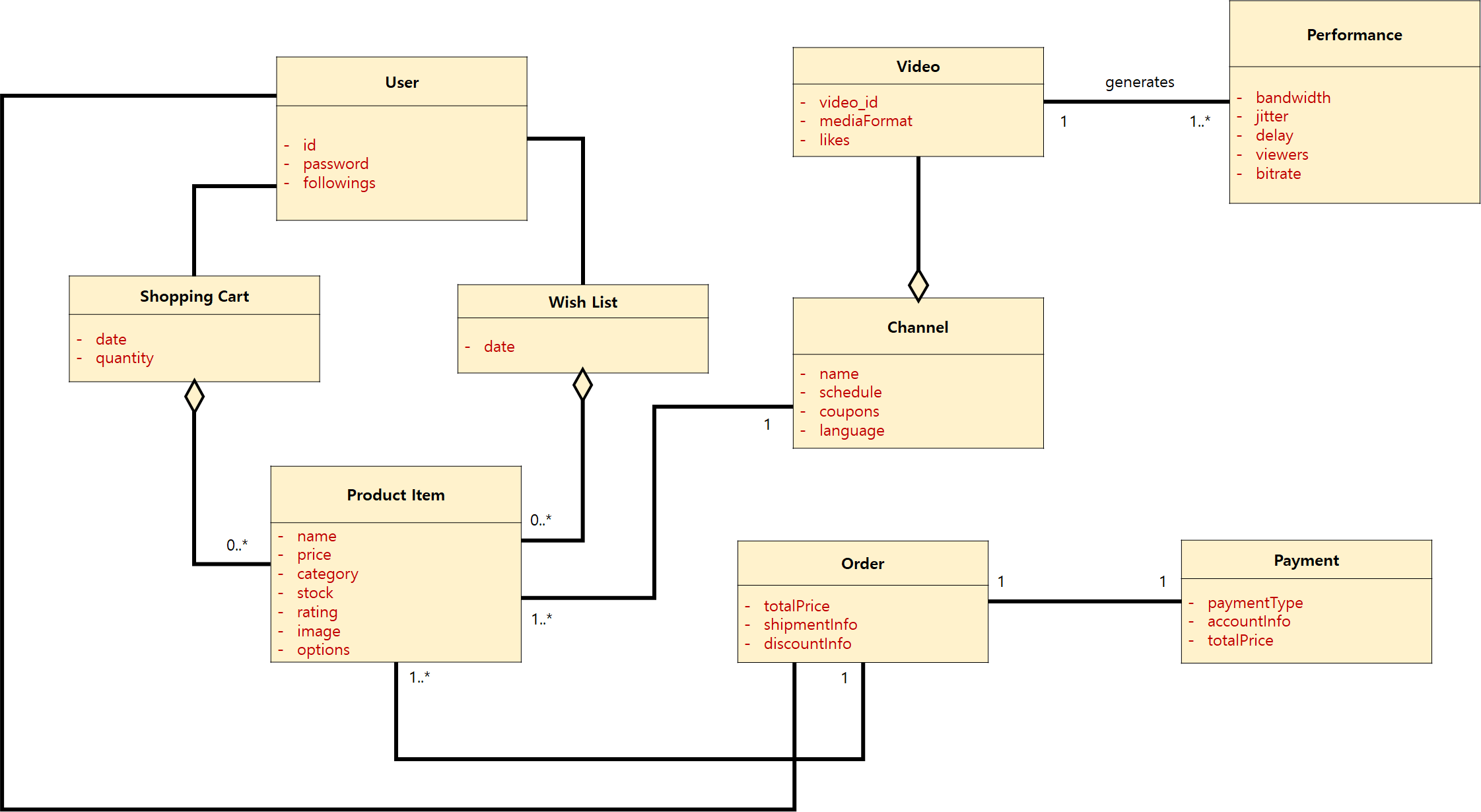
|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale and assumptions** |
| Create a domain model for the application | Before starting functional decomposition, you need to create an initial domain model of the system and identify the main entities in the domain along with their relationships. The domain model should eventually be created. Otherwise, it will appear in a suboptimal way and lead to a temporary architecture that is difficult to understand and maintain. |
| Identify domain model that map to functional requirements | Each individual functional element of the application must be encapsulated in its own contained building block (domain object). One possible alternative is to disassemble the subsystem(server) directly into the module without considering the domain object, but the risk of not considering the requirements increases. |
| Decompose domain objects into general and specialized components | Domain objects represent a complete set of features, but this feature is supported by more granular elements within the subsystem. In the pattern, 'components' are called modules.  The specialization of a module is related to the layer in which it is located. There is no good alternative to break down layers into modules to support functionality. |
| Use JSON RPC to communicate among components | JSON RPC is a remote procedure call protocol encoded in JSON. It's a very simple protocol, but you can organize your data structurally. |

* 1. Instantiate architectural elements, allocate responsibilities, and define interfaces

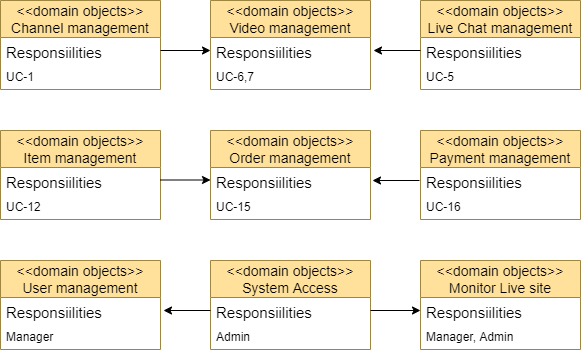
|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale** |
| Generate only initial domain model | Need to identify and model the entities participating in the basic use case, but only the initial domain model is created to accelerate this design phase. |
| Map the system use cases to domain objects | An initial identification of domain objects can be made by analyzing the system’s use cases. To address CRN-3, domain objects are identified for all the use cases in section 3.2. |
| Decompose the domain object in the subsystem to identify the local module with an explicit interface. | This technique ensures that modules that support all the functionalities are identified.  The architect will perform this task just for the primary use cases. This allows another team member to identify the rest of the modules, thereby allocating work among team members.  Having established the set of modules, the architect realizes the need to test these modules, so a new architectural concern is identified here:  CRN-4: A majority of modules should be unit tested.  Only ‘a majority of modules’ are covered by this concern because the modules that implement user interface functionality are difficult to test independently. |
| Use JSON RPC for Interfaces for connecting between components | Interfaces the components across entire sub-systems uses are exported with JSON RPC |

* 1. Sketch views and record design decisions

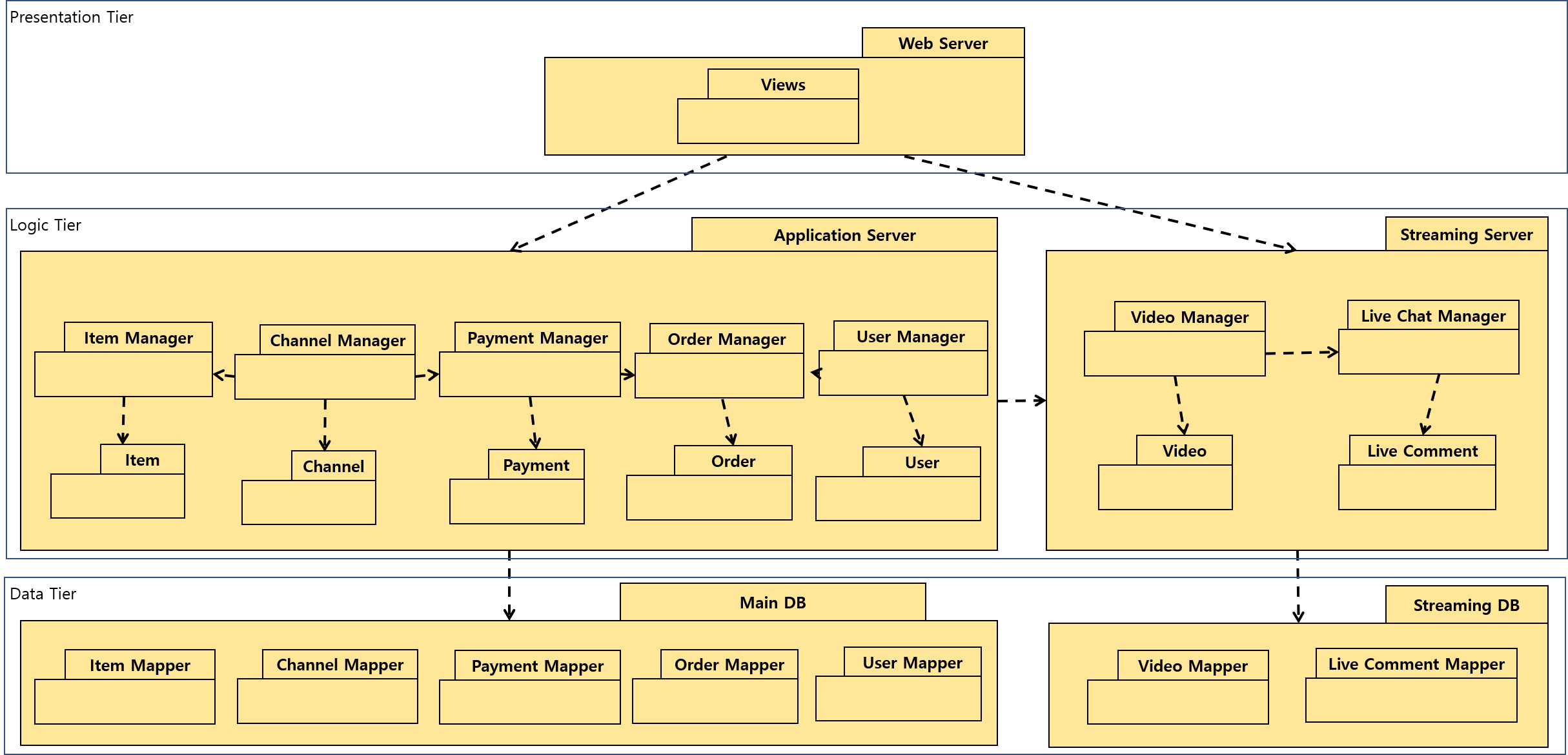
Initial domain model



Domain objects associated with the use case model



Modules that support the primary use cases



|  |  |
| --- | --- |
| **Element** | **Responsibility** |
| Views | Constructs views which users will faces |
| Item Manager | Item Manager is used to perform activities such as Upload item and browse item. |
| Channel Manager | Channel Manager contains modules that controls activities such as open channel, follow channel and enter channel. |
| Payment Manager | Payment Manager where all the payment process will be handled. |
| Order Manager | Order Manager contains modules that activities related with user’s order. |
| User Manager | User Manager contains modules that add/remove/modify user information. |
| Video Manager | Video Manager contains modules that performs Live streaming operation. |
| Live Chat Manager | Live Chat Manager is used to control chat operation during live stream. |
| Entities | Every entity like Item, Channel, etc. are representation of data in DB. |
| Object Mappers | Object Mappers like Item Mapper, Channel Mapper, etc. queries data from DB and maps it to class. |

* 1. Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

|  |  |  |  |
| --- | --- | --- | --- |
| **Not addressed** | **Partially addressed** | **Completed addressed** | **Design decisions Made During the Iteration** |
|  | UC-1 ~  UC-16 |  | Modules across the sub-systems preliminary interfaces to support this use cases have been identified. |
| CRN-4 |  |  | Modules need to be unit tested are not identified |
|  | QA-1 |  | The elements that support the associated use cases have been identified. |
|  | QA-2 |  | The elements that support the associated use cases have been identified. |
|  | QA-8 |  | The elements that support the associated use cases have been identified. |
|  | QA-9 |  | The elements that support the associated use cases have been identified. |

7. Iteration 3 – Addressing Quality Attribute Scenario Driver (QA-5)

* 1. Establish Iteration Goal by Selecting Drivers

Based on the fundamental structural decisions made in iterations 1 and 2, we can now start to reason about the fulfillment of some of the more important quality attributes

* 1. Choose One or More Elements of the System to Refine

For this iteration, the architect focuses on the QA-5 quality attribute scenario:

- streaming delay should be less than 5 seconds.

* 1. Choose One or More Design Concepts that Satisfy the Selected Drivers

The elements that will be refined are the physical nodes that were identified during the first iteration:

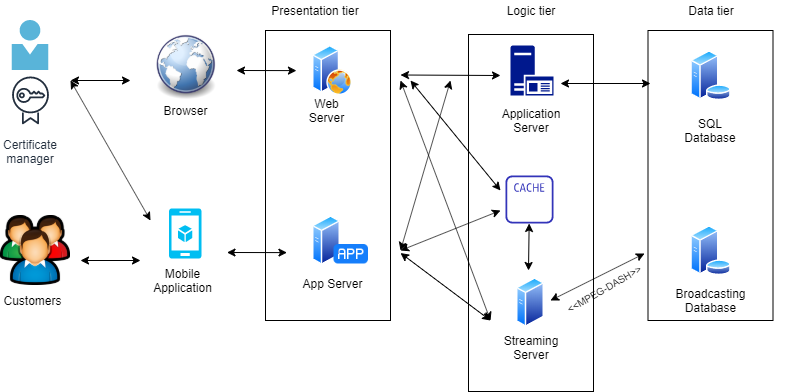
* Streaming Server / Application Server / Web Server
* Streaming database

|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale and assumptions** |
| Introduce **redundancy** tactic by replicating the streaming/application/web server and other critical components such as database | Duplicate critical [components](https://en.wiktionary.org/wiki/component) or functions of a system with the intention of increasing reliability of the [system](https://en.wikipedia.org/wiki/System), usually in the form of a backup or [fail-safe](https://en.wikipedia.org/wiki/Fail-safe), or to improve actual system performance. Using multiple components instead of a single component can increase reliability and availability through redundancy. |
| Introduce the **caching**  tactic by distributing cache memory | By placing previously requested information in temporary storage or cache, the cache server speeds up access to data and reduces demand for corporate bandwidth. |
| Introduce **adaptive bitrate streaming** for streaming data | Adaptive bitrate streaming is a technology used to stream multimedia over computer networks. It works by detecting bandwidth and CPU capacity in real time and dynamically adjusting the quality of media streams. |

* 1. Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

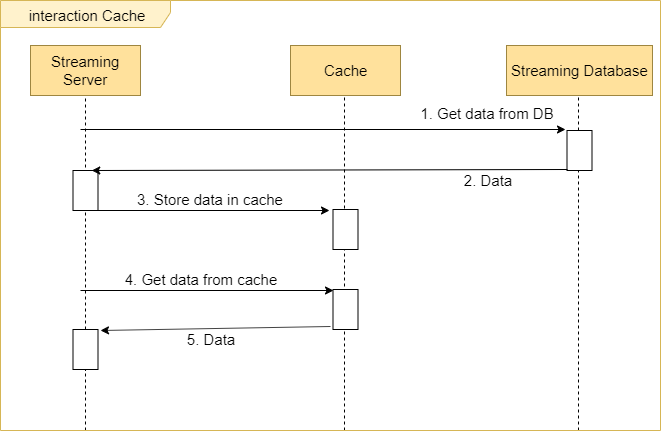
|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale** |
| Use **active redundancy** and **load balancing** in the streaming server, application server, and app server | Selecting active redundancy allows to monitor the performance of individual devices to prevent performance degradation, which is used for voting logic. Voting logic is tied to automatically switching reconfigure components. Multiple servers are active at any time, so we need to distribute and balance the load among them. |
| Use **Cache memory** for streaming data | Choose server side cache to act of caching data on the server. It is common to cache commonly used data from the DB to prevent hitting the DB every time the data is required. |
| Use **MPEG-DASH** protocol for streaming data | It supports adaptive bitrate streaming. Hence viewers will always be delivered the best video quality that their current internet connection speed can support. |
| Use **WOWZA** for streaming engine | Wowza Streaming Engine is powerful custom media server software that lets you reliably stream high quality video and audio to any device, anywhere. Whether we use it in the cloud or in a traditional PC environment, we can leverage the powerful components of the Wowza Streaming Engine to securely coordinate the workflow. |

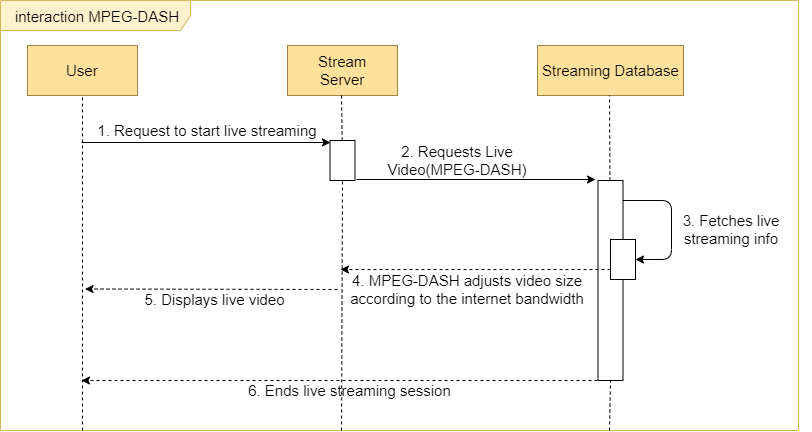
7.5 Sketch Views and Record Design Decisions



|  |  |
| --- | --- |
| **Element** | **Responsibility** |
| CACHE | server side caching is the act of caching data on the server. Data can be cached anywhere and at any point on the server that makes sense. It is common to cache commonly used data from the DB to prevent hitting the DB every time the data is required. |
| MPEG-DASH | It supports adaptive bitrate streaming. That means viewers will always be delivered the best video quality that their current internet connection speed can support. This can fluctuate second-to-second, and DASH can keep up. |

Sequential diagram





* 1. Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

|  |  |  |  |
| --- | --- | --- | --- |
| **Not addressed** | **Partially addressed** | **Completed addressed** | **Design decisions Made During the Iteration** |
|  |  | UC-6,7 | Modules across the sub-systems preliminary interfaces to support this use cases have been identified. |
|  |  | QA-5 | The elements that support the associated use cases have been identified. |
|  | UC-11 |  | Modules across the sub-systems preliminary interfaces to support this use cases have been identified. |
| QA-3 |  |  | No relevant decisions have been made. |

1. Iteration 4 – Addressing Quality Attribute Scenario Driver (QA-1)
   1. Establish Iteration Goal by Selecting Drivers

Goal of iteration 4 is satisfying QA-1. Our system has important database that contain critical information like user information or payment information. We have to protect these information from external. To protect information from attackers, we should improve our system . In this case, we have to concern trade-off between performance and security. We should compromise between performance and security. If we reinforce security, performance will be decrease. It is inevitable

* 1. Choose One or More Elements of the System to Refine

In iteration4, the first quality attribute scenario (QA-1 Security) will be addressed.

* When user use the any function in the system, personal information updated, private payment , purchase history, user personal information, it should be not visible to external. Only to users can access to this.
* When the items of data that are important for customer have been identified, it`s important to decide their representation form.
  1. Choose One or More Design Concepts that Satisfy the Selected Drivers

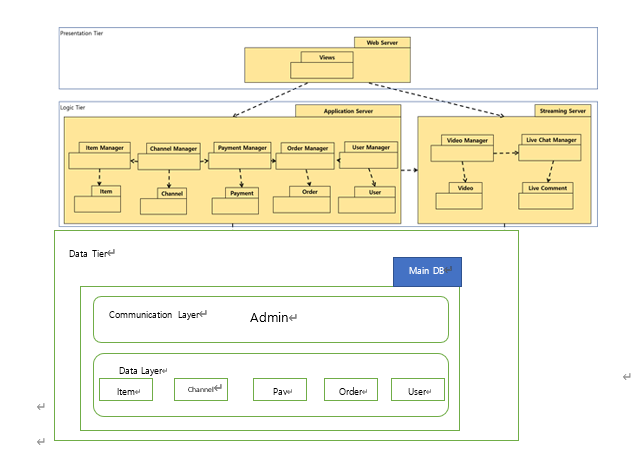
For security, we select layer scheme to control flows of data. There are two flows; one is external flow between server and database, the other is internal flow in database.

|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale and assumptions** |
| Introducing the **layered architecture** to main database for security about critical data such as user account or payment information. | Our database has several mappers (item, channel, payment, order, user,wishlist). If managers in application server communicate with data mappers directly, developer cannot control each flow of data between manager and mapper. These factors increase complexity and vulnerability.  To fix this problem, we choose layered architecture to main database. |

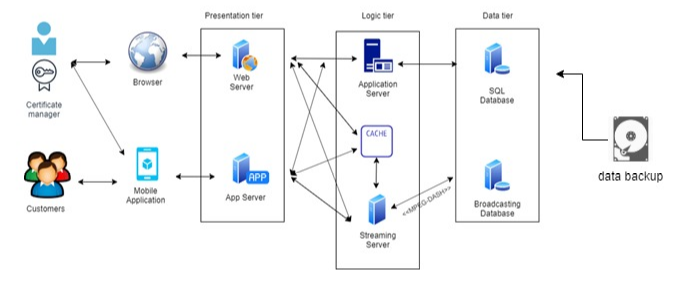
* 1. Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale** |
| Apply DBMS for creating and managing database | A database management system is a system software managing the data of all factors for example user information, items, video url. That can be accessed by multiple users in a controlled manner.  DBMS can help that users to create, read, delete, update through database. It considered as an interface controlling the database by end users, so user can access easily more than  DBMS manages the data which is user information, account, item information, video information. The database schema, which defines the database`s structure, These factors is the important things provide concurrency, security, data integrity like that. |
| The Admin is used for managing and protecting private database. | ***Admin*** is a manager of the data in the database, rather than a user.  Admin has responsibility for the development, implementation, operation, maintenance and security of the database and the applications that use it.    Admin  knows about whole development of the database structure and data dictionary  the provision of security measures to permit authorized access and prevent unauthorized access to data  to prevent failures in hardware or software in order to offer reliability. |
| Database Backup is also important for enhancing the security | Database backup is the process of store the operational state architecture and whole data about the system. It copy the creation of a instance about the whole system. If it attacked from external or other reason, we can recover the easily whole process |

* 1. Sketch Views and Record Design Decisions



**Deployment Diagram**



Interaction between application server and main Database will be encrypted. It could be vulnerable for external intrusion. We need to prevent many threats. So we divide the server according to the function, and back up the data using other device.

* 1. Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

|  |  |  |  |
| --- | --- | --- | --- |
| **Not addressed** | **Partially addressed** | **Completed addressed** | **Design decisions Made During the Iteration** |
|  |  | All case | All use case model should be encrypted and protected |
|  | CRN-2  CRN-3 |  | These architectural concern are introduced but nor relevant so much. |
|  |  | CRN-1 | When establish an overall system architecture we consider this factor mostly |
|  |  | QA-2 | The elements that support the associated use cases have been identified. |
| QA-3,QA-6 |  |  | Not considered. |
|  |  | QA-1 | The elements that support the associated use cases have been identified. It is main attribute in this part |
|  |  | CON-2 | Customer`s data is relevant this factors |

1. Iteration 5 – Addressing Quality Attribute Scenario Driver (QA-8)
   1. Establish Iteration Goal by Selecting Drivers

Our system is based on e-commerce system, payment system is the most important function of the in our system. So, Keeping safety of the this system without failure also important in payment process and architectural design decision. In this iteration, we concern about the payment safety attributes

* 1. Choose One or More Elements of the System to Refine

In iteration 5, we will cover quality attribute Safety

1. When users process a payment for their items, bekoor site and payment system should be safe.
2. The whole process must be done in one transaction in safety circumstance.
   1. Choose One or More Design Concepts that Satisfy the Selected Drivers

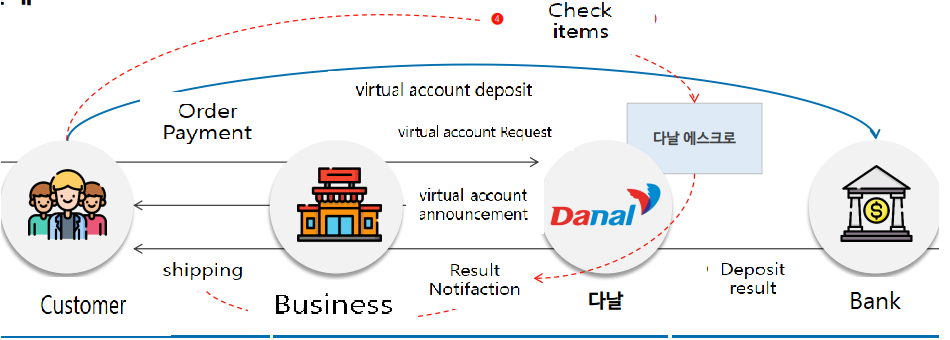
|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale and assumptions** |
| Introducing ”payment gateway” | A payment system service provided by an payment cooperation Danal that authorizes credit card or mobile phone payment or account transfer for e-businesses, online retailers. The payment gateway may be provided by Danal. A payment gateway facilitates a payment transaction among users, card cooperation, telecommunications company, bank. |

* 1. Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

We improve the safety adding the real time rm (risk Management) system to prevent illegal transaction, fraudulent transaction.

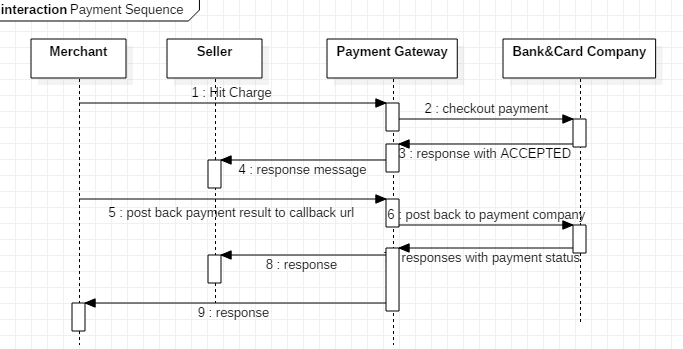
|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale** |
| Introducing “RM” architecture for secure payment transaction system. | -To protect the payment system, there is an rm  Rm is the risk management real time system that protect the payment process.  -This system observe the dangerous factors and block the factors like illegal transaction or fraudulent transaction analyzing the payment pattern using high quality data statistics techniques. |
| Checking the process of the RM architecture. | To know about the whole process about the rm system which is the protect system about the payment gateway  First through monitoring system, recognize the defect  Second Check the defect type  Third Notify the information to business  Fourth The person in charge takes first aid.  Fifth After the fix, report the contents about defect and fix to business. |

* 1. Sketch Views and Record Design Decisions



|  |  |
| --- | --- |
| **Element** | **Responsibility** |
| Danal | Danal is the payment gateway. Danal provide the virtual account and announce about virtual account and notify the result after the deposit.  Customer can check items and order`s will completely finished. |
| Customer | Customer order the items and deposit to virtual account. And items will be shipped and customer get the items.  After checking items ,payment is delivered to business and order process is finished. |

Below is payment sequence diagram. The payment gateway performs control and monitoring of the payment process as a whole and responds to the occurrence of an error situation during the transaction.



* 1. Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

|  |  |  |  |
| --- | --- | --- | --- |
| **Not addressed** | **Partially addressed** | **Completed addressed** | **Design decisions Made During the Iteration** |
|  |  | UC-16 | It is all about the checkout payment so it is completely related . |
|  | CRN-2 |  | This new architectural concern is introduced in this iteration: no relevant decisions have been made. |
|  | CRN-3 |  | This new architectural concern is introduced in this iteration: no relevant decisions have been made. |
|  |  | CRN-1 | We consider the this system from the start, buy and pay are the most important behavior |
|  |  | QA-8 | The elements that support this Quality Attributes completely. |
| QA-4,QA-5 |  |  | No relevant decisions have been made. |

1. Iteration 6 – Addressing Quality Attribute Scenario Driver (QA-3)
   1. Establish Iteration Goal by Selecting Drivers

Business goal of our system is to stimulate users' desire to purchase using live streaming. If user can access to our system using various devices (PC and mobile), we can get more chance to make profit. In other word, we should support PC website and mobile application (or website) both. At the same time, we should not change original architecture too much while add some modules that support portability. Therefore, our goal in iteration 6 is modifying the architecture to support usability (QA-3) and support as many devices as possible which is of Android operating system (CON-3).

* 1. Choose One or More Elements of the System to Refine

In iteration 6, quality attribute scenario 3 (QA-3: portability) will be discussed.

1. Bekoor 6 should be produced for different computing platforms, such as Mobile APP, Mobile web browser and Desktop web browser.
2. Also, Bekoor 6 should be able to move across different environment, not just across platforms. For example, modify software and make it adaptable to work on Linux, Windows, Mac OS, Unix, etc.

Hence, the system interfaces and application logic will be analyzed respectively.

* 1. Choose One or More Design Concepts that Satisfy the Selected Drivers

|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale and assumptions** |
| Using “Cross-platform” development tools | Following the classic native approach, developers produce two different applications: one written in Java for Android and the other in Swift or Objective-C for iOS. This path is time-consuming and expensive because the same code must be written twice.  Cross-platform development, allows engineers to write the code once and apply some parts of it across all platforms. |
| Introducing “Factory patterns” | We choose factory pattern to avoid modifications to the existing core functionality. Factory pattern describe how to solve recurring design problems to design flexible and reusable object-oriented software, that is, objects that are easier to implement, change, test, and reuse. |

* 1. Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale** |
| React Native to build cross-platform mobile applications | React Native uses JavaScript, currently, one of the most popular high-level, dynamic programming languages.  React Native also combines the benefits of JavaScript and React.JS, a web framework, and is backed by Facebook. The strong side of React Native is that it allows for writing modules in Objective-C, Swift, or Java languages when developers need them. Using native modules and libraries in React Native apps, you can handle computationally heavy operations like image editing and video processing or some unconventional operations that aren’t included in the framework APIs. |
| Design Java **factory patterns** for class-based programming. | The factory patterns in Java uses factory methods to deal with the problem of creating objects without having to specify the exact class of the object that will be created.  \* For easier understanding, we assume Bekoor 6 contains multiple live streaming videos, and each video seller sells products, then the relationship between the three is abstract factory (Bekoor 6), simple factory (video seller), and product. |

* 1. Sketch Views and Record Design Decisions

**지도, 텍스트, 하늘, 실내이(가) 표시된 사진

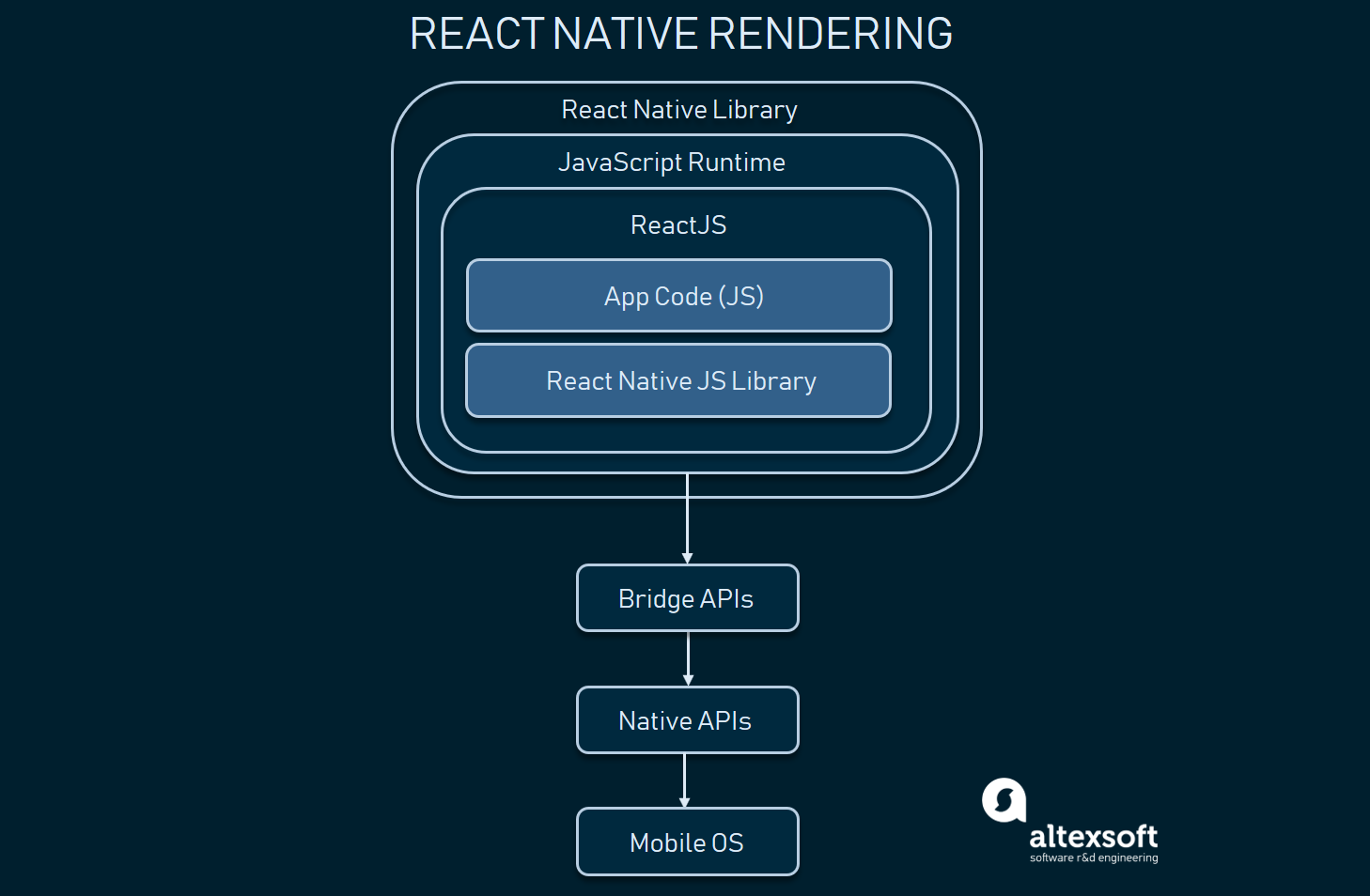
자동 생성된 설명**

**React Native: rendering to the native APIs**

React Native performance is close to native because it renders code components directly to the native APIs using the [JavaScriptCore](https://trac.webkit.org/wiki/JavaScriptCore) virtual machine both for iOS and Android. While JavaScriptCore is included in iOS by default, React must install the VM to Android, which slightly increases the Android app size. React also allows for using native iOS and Android modules written in Objective-C and Java. They can’t be reused across the two platforms and their main purpose is achieving tasks that aren’t included in cross-platform JavaScript APIs. They also may be helpful in achieving higher performance in complicated operations like image editing or video playback. To connect with native modules you have to build so-called bridge APIs both for [Android](https://facebook.github.io/react-native/docs/native-modules-android.html) and for [iOS](https://facebook.github.io/react-native/docs/native-modules-ios.html). As we mentioned, these require understanding native platforms to fully leverage the framework.

The rest of the codebase can be shared across platforms. Generally, React Native applications can share from 80–90 percent of JS code. Using native libraries also optimizes app performance.

React Native uses JIT compilation for Android but it doesn’t provide AOT for iOS. In case of iOS, it doesn’t compile but interprets the JavaScript code.



* 1. Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

|  |  |  |  |
| --- | --- | --- | --- |
| **Not addressed** | **Partially addressed** | **Completed addressed** | **Design decisions Made During the Iteration** |
|  | All relative  UC |  | This new architectural concern is introduced in this iteration: no relevant decisions have been made. |
|  |  | CON-1 | The streaming screen should be compatible to various size and brand of mobile phone. |
|  | CON-3 |  | The system should work on every Android phone. |
|  |  | QA-3 | The elements that support the associated use cases have been identified. |

1. Iteration 7

security driver for web management system QA-9 and

even though web management system is a simple web page, it is the place where roles and contents of the whole service is configured. We consider authentication as the most important desire for this part. The last iteration addresses the security driver of the web management system.

11.1 Establish Iteration Goal by Selecting Drivers

|  |  |
| --- | --- |
| QA-9 | a user log in the web service. authentication  function should be performed to make sure that this user is an admin or manager. |
| QA-10 | provide proper view for management like dashboard view templates etc. |

11.2Choose One or More Design Concepts that Satisfy the Selected Drivers

elements that have influence on security and over all architecture of the web management system.

* user
* role
* access of server
* DB which stores role configuration information

11.3 Choose One or More Design Concepts that Satisfy the Selected Drivers

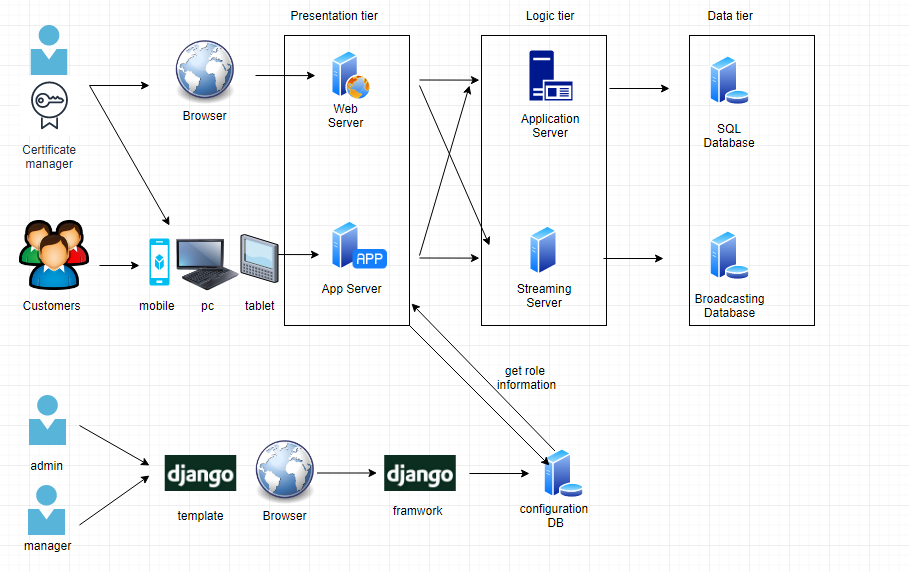
in this step, authentication solutions for admin management should be selected to make sure that only users with the right role (manager/ admin) can access to the management web page. And at the same time all configuration information including contents start and end date, role information should be saved in the DB. This config DB should be connected with application server addressed before.

We decided to use the Django framework which provides the various templates and functionalities of admin management using a traditional MVC pattern.

11.4 Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

|  |  |
| --- | --- |
| **Design decisions and location** | **Rationale** |
| A framework of management system – Django | the Django framework which provides the various templates and functionalities of admin management using a traditional MVC pattern |
| Configuration DataBase | Stores start and end date of the contents (sessions) and role configuration information. This configuration DB could be accessed by application server . |
| Web page - view | Presents the web page for manager and admin users to show session config page and role config page. |

11.5 Sketch Views and Record Design Decisions



11.6 Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

|  |  |  |  |
| --- | --- | --- | --- |
| **Not addressed** | **Partially addressed** | **Completed addressed** | **Design decisions Made During the Iteration** |
|  | QA-9 |  | a user log in the web service. authentication  function should be performed to make sure that this user is an admin or manager. |
|  |  | QA- 10 | provide proper view for management like dashboard view templates etc. |