

# Fitts: Recommendation System with ADD

(Final Project Report)

Software Engineering Technologies

(ECE596641)

Team2

NAME	STUDENT ID
빙 (Nguyen Khanh Binh)	2019711305
임근식 (Lim Geunsik)	2019711699
하지크 (Haziq Hamzah)	2019711435

## 목차

Revision History .....	5
Objectives .....	6
1. Overview of the Target SYstem .....	7
1.1. Fitts .....	7
1.2. Business case .....	8
2. System Requirement .....	9
2.1. Use case model .....	9
2.2. Quality attribute scenarios .....	11
2.3. Constraints .....	12
2.4. Architectural concerns .....	12
3. Review Inputs .....	13
4. Design Process: Iteration 1 (System Architecture) .....	15
4.1. Step 2: Establish iteration goal by selecting drivers .....	15
4.2. Step 3: Choose one or more elements of the system to refine .....	15
4.3. Step 4: Choose one or more design concepts that satisfy the selected drivers .....	15
4.4. Step 5: Instantiate architectural elements, allocate responsibilities, and define interfaces .....	17
4.5. Step 6: Sketch views and record design decisions .....	19
4.6. Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose .....	20
5. Design Process: Iteration 2 (Selecting Technologies) .....	21
5.1. Step 2: Establish iteration goal by selecting drivers .....	21

5.2. Step 3: Choose one or more elements of the system to refine .....	21
5.3. Step 4: Choose one or more design concepts that satisfy the selected drivers .....	21
5.4. Step 5: Instantiate architectural elements, allocate responsibilities, and define interfaces .....	22
5.5. Step 6: Sketch views and record design decisions .....	23
5.6. Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose .....	27
6. Design Process: Iteration 3 (Quality Attributes Scenarios Drivers) .....	28
6.1. Step 2: Establish an iteration goal by selecting drivers .....	28
6.2. Step 3: Choose one or more elements of the system to refine .....	29
6.3. Step 4: Choose one or more design concepts that satisfy the selected drivers .....	29
6.4. Step 5: Instantiate architectural elements, allocate responsibilities, and define interfaces .....	29
6.5. Step 6: Sketch views and record design decisions .....	30
6.6. Step 7: Perform analysis of current design and review iteration goal and achievement of design purpose .....	31
7. Design Process: Iteration 4 (Constraints Drivers) .....	32
7.1. Step 2: Establish Iteration Goal by Selecting Drivers .....	32
7.2. Step 3: Choose One or More Elements of the System to Refine .....	32
7.3. Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers .....	32
7.4. Step 5 and 6: Instantiate Architectural Elements, Allocate Responsibilities, Define Interfaces and Sketch Views and Record Design Decisions .....	32
7.5. Step 7: Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose .....	33
8. Design Process: Iteration 5 (Architectural Concern Drivers) .....	34

8.1. Step 2: Establish Iteration Goal by Selecting Drivers .....	34
8.2. Step 3: Choose One or More Elements of the System to Refine .....	34
8.3. Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers .....	34
8.4. Step 5 and 6: Instantiate Architectural Elements, Allocate Responsibilities, Define Interfaces and Sketch Views and Record Design Decisions .....	35
8.5. Step 7: Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose .....	35
9. Conclusion .....	36
Appendix. Terminology .....	37
Appendix. Kanban Board .....	38
Appendix. GitHub Repository .....	39
Appendix. References .....	40

## REVISION HISTORY

We summarize the revision history on this document as follows.

Version	Date	Author	Description
0.10	Oct-16-2019	All	Overview of target system, Business Case
0.20	Oct-21-2019	All	System Requirement
0.30	Nov-23-2019	All	Review Inputs, Create Kanban board with GitHub
0.40	Nov-28-2019	All	Write ADD drivers
0.60	Dec-01-2019	임근식	Iteration 1
0.65	Dec-01-2019	임근식	Iteration 2
0.70	Dec-04-2019	하지크	Iteration 3
0.75	Dec-05-2019	빙	Iteration 4
0.90	Dec-06-2019	빙	Iteration 5
0.95	Dec-08-2019	All	Review and update the iteration 1, 2, 3, 4, and 5 (Iteration Revision v1.0)
0.96	Dec-09-2019	All	Review and update the iteration 1, 2, 3, 4, and 5 (Iteration Revision v2.0)
0.97	Dec-10-2019	All	Review and update the iteration 1, 2, 3, 4, and 5 (Iteration Revision v3.0)
1.00	Dec-11-2019	All	Release the final version. The documents are uploaded into the bulletin board of (1) iCampus and (2) the “doc” folder Team2 GitHub repository.

## OBJECTIVES

We describe our recommendation system using ADD 3.0 for a **greenfield system** for a mature domain, such as recommendation of the on-line shopping malls.

- ✓ The domain is still relatively new and quickly evolving.
- ✓ So, the architects could not absolutely rely on the past experience to guide developers and users.
- ✓ We propose the design process with a customer data-based training and inferencing mechanism based on cloud.

Table 1. Greenfield VS. Brownfield system

S. No.	Aspect	Greenfield	Brownfield
1	Project direction	Vague	Clear
2	Development effort	Comparatively more since everything needs to be built from scratch	Comparatively less since basic foundation is already built
3	Dependency on older systems	No	Substantial
4	Development time	Comparatively more	Comparatively less
5	Degree of risk	Comparatively higher	Comparatively lower
6	Re-engineering required	No	Likely
7	Costs	Can be costly if there is no clear direction	Can be costly due to the presence of legacy code

**Greenfield software development** refers to developing a system for a totally new environment and requires development from a clean slate – no legacy code around. It is an approach used when you're starting afresh, and with no restrictions or dependencies. A pure greenfield project is quite rare these days; you frequently end up interacting or updating some amount of existing code or enabling integrations. Some examples of greenfield software development include: building a website or app from scratch, setting up a new data center, or even implementing a new rules engine.

**Brownfield software development** refers to the development and deployment of a new software system in the presence of existing or legacy software systems. Brownfield development usually happens when you want to develop or improve upon an existing application, and compels you to work with previously created code. Therefore, any new software architecture must consider and coexist with systems already in place – so as to enhance existing functionality or capability. Examples of brownfield software development include: adding a new module to an existing enterprise system, integrating a new feature to software that was developed earlier, or upgrading code to enhance functionality of an app.

## 1. OVERVIEW OF THE TARGET SYSTEM

### 1.1. FITTS

This section briefly describes the target system.

#### Background

Nowadays, due to the enormous growing of the internet and technologies, it enables people to do many things through their mobile phone, laptop, television, and Online shopping mall appeared by the growing of both the fashions companies and internet. Mobile shopping mall application provides functions for users convenient shopping. Customers can check details of the product at their home, pay as they go and receive the product at their chosen place.

#### Fitts

While so many existing online shopping mall applications on application store, they are all the same with each other's. Because of that, some company choose to make the different by some minor changes. Fitts is one of those. **The key idea of Fitts success is they using the users body information to suggest the items to users, and by that increase the revenue from each user.**

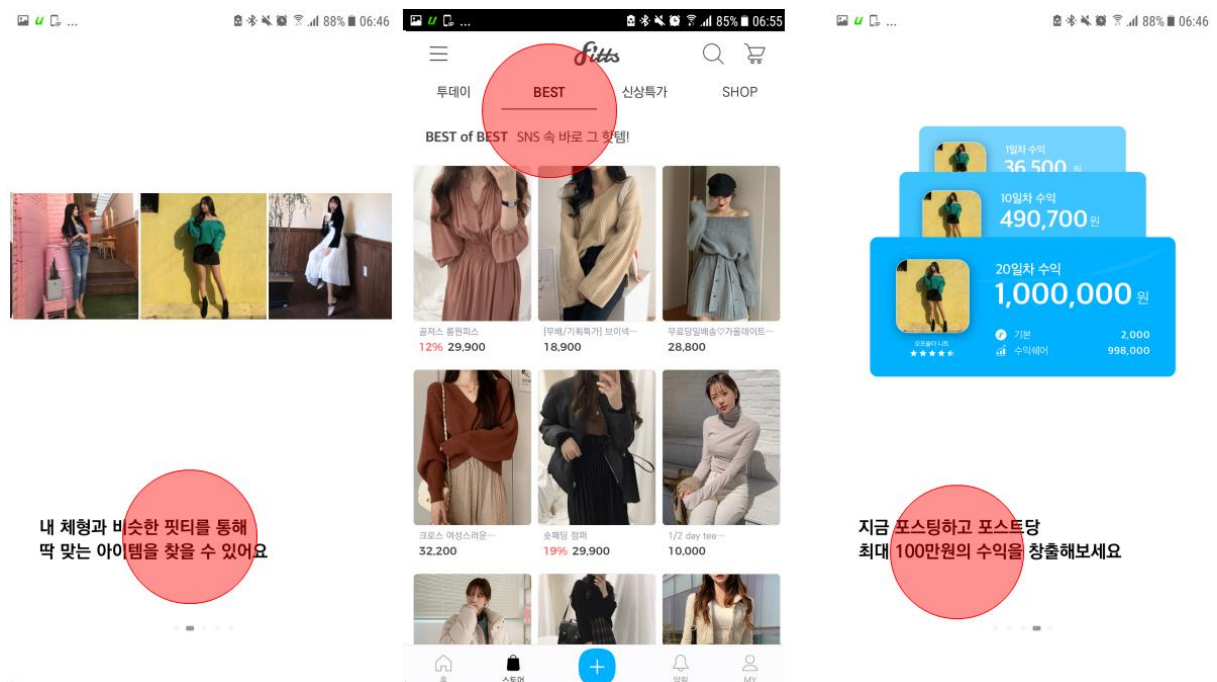


Figure 1. Major functions of the Fitts system

## 1.2. BUSINESS CASE

- An on-line shopping mall companies collect lots of data using cloud.
- Then, they analyze the collected data by introducing the machine learning.
- The company provides customized data and intelligent recommendation service to the millions of the mobile app users.
- Since the AI-based item recommendation system can give the application users more choices, the internet technology department of the company try to introduce cloud and AI based recommendation service.
- Besides, requests for a new system are coming from the customers. They would like to get the user experience in advance before purchasing the items because the new system can combine the user photo and recommended clothes.

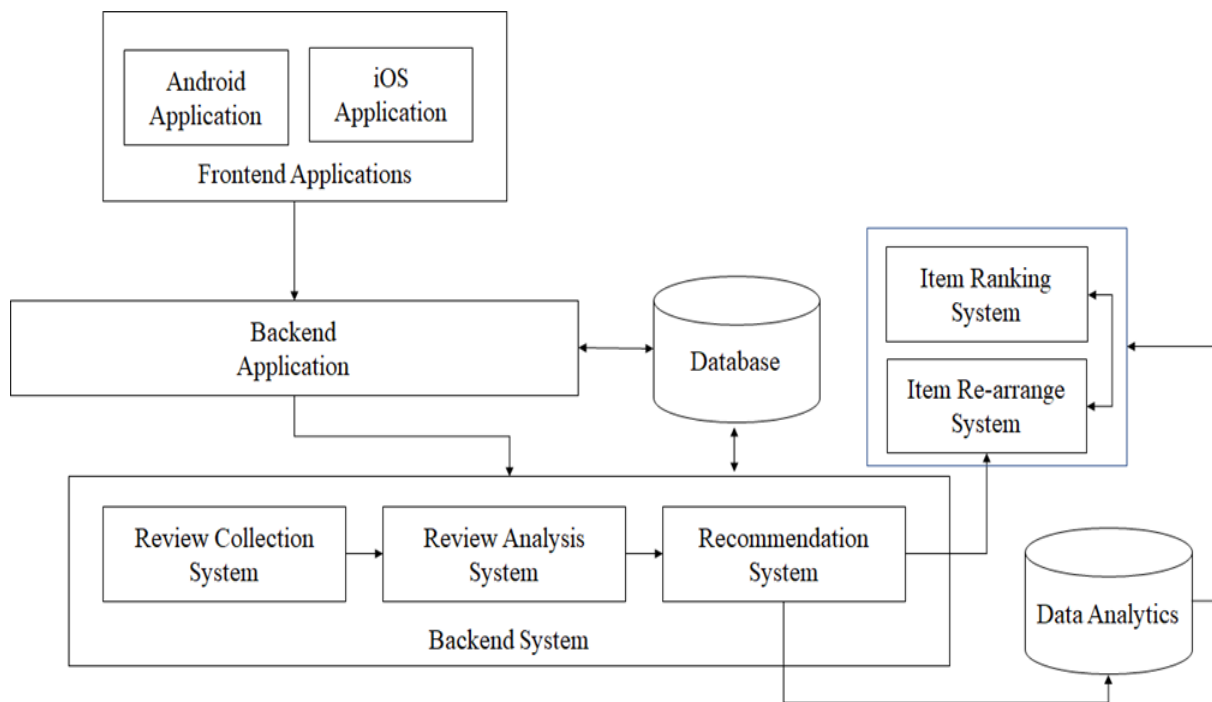


Figure 2. Overall system architecture



## 2. SYSTEM REQUIREMENT

### 2.1. USE CASE MODEL

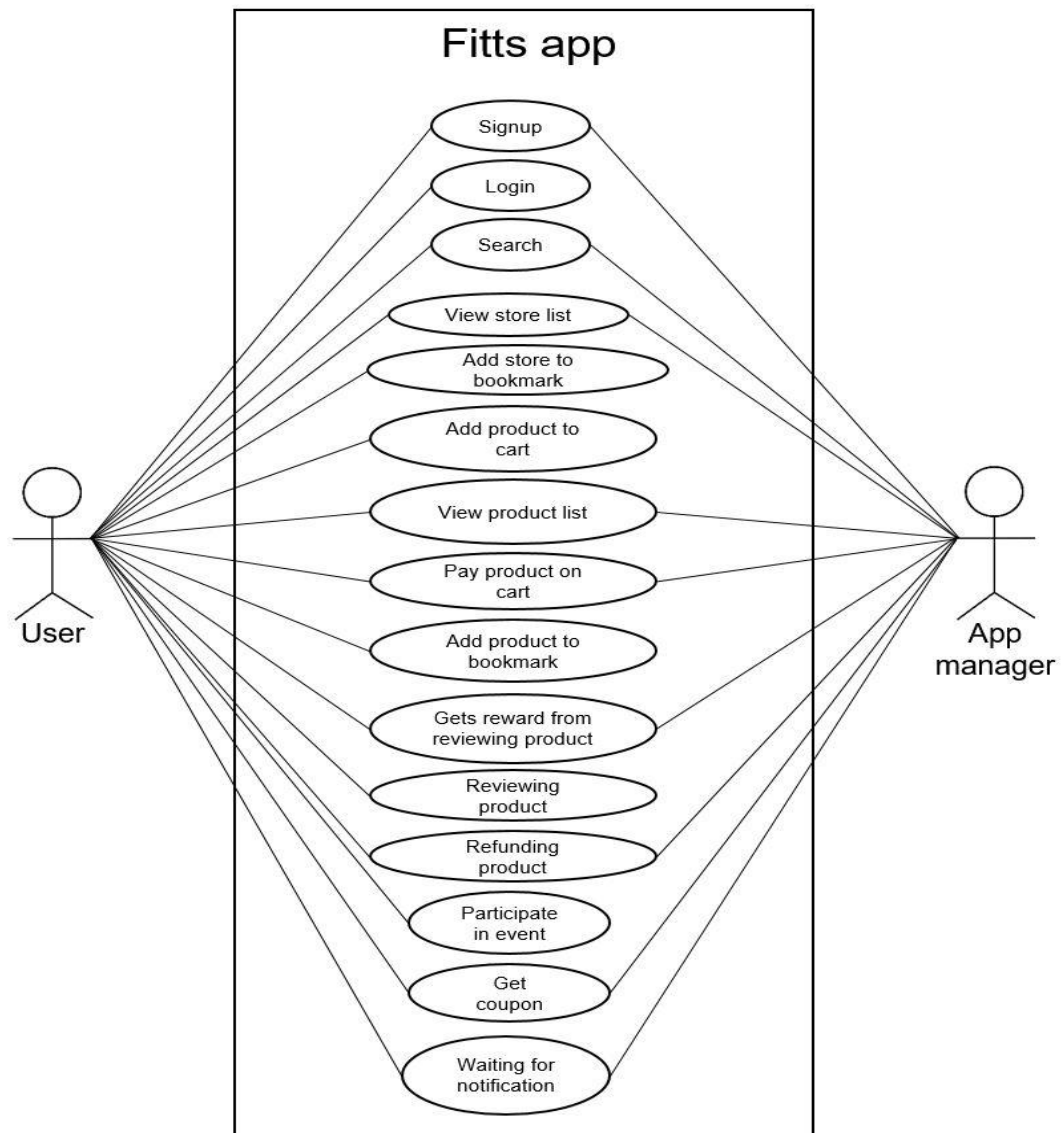


Figure 3. Use-case diagram

Use case	Description
<b>UC-1: Signup</b>	The user sign-up to the system when first using it while app manager provides a new account for the user
<b>UC-2: Login</b>	The user log into the app using his/her account
<b>UC-3: Search</b>	User search for some keywords about product or store. The app manager will provide the information related the keywords searched by user
<b>UC-4: View store list</b>	User view the list of stores available in the app database. The app manager responsible to organize the data for better user experience
<b>UC-5: Add store to bookmark</b>	User add a store in his/her bookmark
<b>UC-6: Add product to cart</b>	User add a product into his/her cart
<b>UC-7: View product list</b>	User view the list of products available in the app database. The app manager responsible to organize the data for better user experience
<b>UC-8: Pay product on cart</b>	User make payment of products in his/her cart. The app manager confirms the product payment and notice user about it
<b>UC-9: Add product to bookmark</b>	User add a product to his/her cart
<b>UC-10: Gets reward from reviewing product</b>	User gets reward (Fitts point) by reviewing products. The app manager makes sure user get the right amount of Fitts point
<b>UC-11: Reviewing product</b>	User review products
<b>UC-12: Refunding product</b>	User request for product refund. The app manager must approve or disapprove the refund request from user
<b>UC-13: Participate in event</b>	User participate in events available in the Fitts app
<b>UC-14: Get coupon</b>	User request for coupon. The app manager approves or disapprove the request
<b>UC-15: Waiting for notification</b>	User waits for notification about payment confirmation or events. The app manager sends the notices about payment confirmation or events

## 2.2. QUALITY ATTRIBUTE SCENARIOS

ID	Quality attribute	Scenario	Associated use-case
QA-1	Performance	The app view refresh time when receiving the user's request must be less than 1 second.	UC-4, UC-7
QA-2	Performance	Time for loading the data should be less than 1 second.	UC-3, UC-4, UC-7
QA-3	Security	Because of the point system working as the real money system, so it needs the security level like the banking system.	UC-8, UC-10, UC-14
QA-4	Scalability	System must be large enough to adapt to the huge amount of user data every day, mostly for image data which costs a lot of space.	UC-1, UC-5, UC-6, UC-9
QA-5	Availability	The system shall continue to operate with no downtime if any single node or component fails	All use cases
QA-6	Deployability	The system deployment procedure shall be fully automated and support a number of environments: development, test, and production	All use cases

## 2.3. CONSTRAINTS

ID	Constraint
CON-1	Fitts point must not change unintentionally without following the user's activity (ex. shopping with Fitts point)
CON-2	No third-party access is allowed to user's Fitts database
CON-3	An existing relational database server must be used. This server cannot be used for other purposes than hosting the database
CON-4	A minimum of 100 simultaneous users must be supported
CON-5	The system must be reliable, due to the number of actions will be performed between user and application server; the system must support some authorization in every API calls
CON-6	The system issues the RSA-based encrypted keys when the system has to publish a token key for new users

## 2.4. ARCHITECTURAL CONCERNS

ID	Concern
CRN-1	Must establish an understanding about the overall system structure
CRN-2	Allocate work to members of the development teams
CRN-3	Leverage the team's knowledge about core technologies using in the system
CRN-4	The elements of the system have to be interconnected with a high bandwidth such as 10Gbps

### 3. REVIEW INPUTS

Basically, we follow the roadmap of the design for greenfield systems in mature domains. In order to enhance the recommendation structure of the existing recommendation system, we improve the mature domains by modifying some uncertainties inherent in the recommendation domain.

Table 2. Design purpose and functional requirements

Category	Details
<b>Design purpose</b>	This is a greenfield system in a mature domain. The major purpose is to design for the next system release of Fitts recommendation system. The organization will perform development following an Agile process with iteration procedure so that developers can quickly receive customer feedback and continue modifying the system.
<b>Primary functional requirements</b>	The primary use case for this release is UC-7, UC-10, UC-11, and UC-14.

Table 3 Quality attribute scenarios

Scenario	Importance to customer	Difficulty of implementation according to architect
QA-1	High	Medium
QA-2	Medium	Medium
QA-3	High	Medium
QA-4	Medium	Medium
QA-5	Medium	High
QA-6	High	High

Table 4. Constraints and Concerns

Category	Details
<b>Constraints</b>	<ol style="list-style-type: none"> <li>1. We have to support the reliable recommendation system with a robust security facility when the recommendation system analyzes the review contents based on the deep learning system.</li> <li>2. The system issues the RSA-based encrypted keys when the system has to publish a token key for new users.</li> </ol>

3. An existing relational database server must be used. This server cannot be used for other purposes than hosting the database.
4. A minimum of 100 simultaneous users must be supported
5. The system must be reliable, due to the number of actions will be performed between user and application server; the system must support some authorization in every API call.
6. Fitts point must not change unintentionally without following the user's activity (ex. shopping with Fitts point). No third-party access is allowed to user's Fitts point database

**Architectural concerns**

1. When the system will be proliferated to others (e.g. shoots, animals, accessories such as a ring, a necklace, and bags as well as the existing clothes), the recommendation system must be compatible with that.
2. The elements of the system have to be interconnected with a high bandwidth such as 10Gbps.
3. We must allocate work to members of the development team, the design team.
4. We must leverage the team's knowledge about core technologies using in the system (Java, Kotlin, Android, ADB, and so on).
5. Python and C++ basic is compulsory for developers to be able to use TensorFlow.

## 4. DESIGN PROCESS: ITERATION 1 (SYSTEM ARCHITECTURE)

In the iteration1, we depict **the reference architecture and overall system structure as a goal of this step.**

### 4.1. STEP 2: ESTABLISH ITERATION GOAL BY SELECTING DRIVERS

This is the 1<sup>st</sup> iteration in the design of a greenfield system, so the iteration goal is to establish an initial overall structure of the system (CRN-1).

Even though this 1<sup>st</sup> iteration is driven by a general architectural concern, the architect has to keep in mind all of the drivers. Especially, the architect must handle constraints and quality attributes:

- CN-1: Do not change the point with another purpose without the user's activity such as shopping the items in the Fitts mall.
- CN-2: Third-party cannot access to the point table of Fitts database.
- CN-3: Support relational database for handling the data dependency and relationship between the item and the review.
- CN-4: Support reliable user responsiveness while using the personalized shopping mall. (The number of the simultaneous users: 100)
- CN-5: Provide reliable security while transferring the data between user and application server with secured API calls.
- CN-6: Establish robust security policy with the RSA-based encrypted keys for user's token key.
- QA-1,2: Performance, QA-3: Security, QA-4: Scalability, QA-5: Availability, QA-6: Deployability

### 4.2. STEP 3: CHOOSE ONE OR MORE ELEMENTS OF THE SYSTEM TO REFINE

Now that our system is the greenfield development and this step is 1<sup>st</sup> iteration step, the element to refine is the entire system.

### 4.3. STEP 4: CHOOSE ONE OR MORE DESIGN CONCEPTS THAT SATISFY THE SELECTED DRIVERS

In this iteration, design concepts are selected from a group of deep-learning based data analytics reference architectures

A design concepts catalog that includes many of the reference architectures. Smart Decisions (<http://smartdecisionsgame.com>) can be played by software architects, software and embedded engineers, data scientists, students, and anyone who is interested in learning about software

architecture design techniques and how they can be applied in the design of software systems with modern technologies like cloud and Machine Learning.

Table 5. Design decisions and location

Design decisions and location	Rationale
<b>Build the application as an instance of the reference architecture</b>	The reference architecture, shown in Fig 1, is a reference architecture that splits the processing of a data stream into two streams: the “Item ranking and rearrange system”, which supports access to real-time data (UC-10, UC-11), and the “Data analytics” system, which support access to cloud data (UC-3, UC-7). But this is different from the item recommendation structure of the existing system of Fitts. While the user application is based on user’s input data, the recommendation system is based on the data analytics techniques with deep-learning to support more intelligent and customizable recommendation service to give users personalized shopping system.
<b>Deep-learning based fitting personalization for all elements in the recommendation system</b>	<p>Now that the AI-based item recommendation system can give the application users more choices, the internet technology department of the company try to introduce cloud and AI based recommendation service.</p> <p>Besides, requests for a new system are coming from the customers. They would like to get the user experience in advance before purchasing the items because the new system (“Customized fitting service”) can combine the user photo and recommended clothes.</p> <p>We need to make sure, in all subsequent design and deployment decisions, that all candidate technologies will support the QA-6 requirement by providing the advanced deep-learning network models (e.g., Mobile SSD Net, Faster RCNN, and Yolo Net) to the “Customized fitting service” principle.</p>

Table 6. Alternatives and reasons for discarding

Design decisions and location	Rationale
<b>Traditional relational</b>	This reference architecture is based on the traditional relational database model principles and SQL-based DBMSs, which are considered highly efficient for complex data queries. However, these DBMS designs cannot guarantee a fast user responsiveness and real-time processing in case that user requests the complicated data queries.
<b>Extended relational</b>	Even though this reference architecture is completely based on relational model principles and SQL-based DBMS, it can



	be improved by parallelized task operations and effective load-balancing mechanism. Also, the deep-learning based item recommendation system can be accelerated by introducing In-memory DBMS and Key-value store DBMS as well as SQL DBMS.
--	---

**4.4. STEP 5: INSTANTIATE ARCHITECTURAL ELEMENTS, ALLOCATE RESPONSIBILITIES, AND DEFINE INTERFACES**

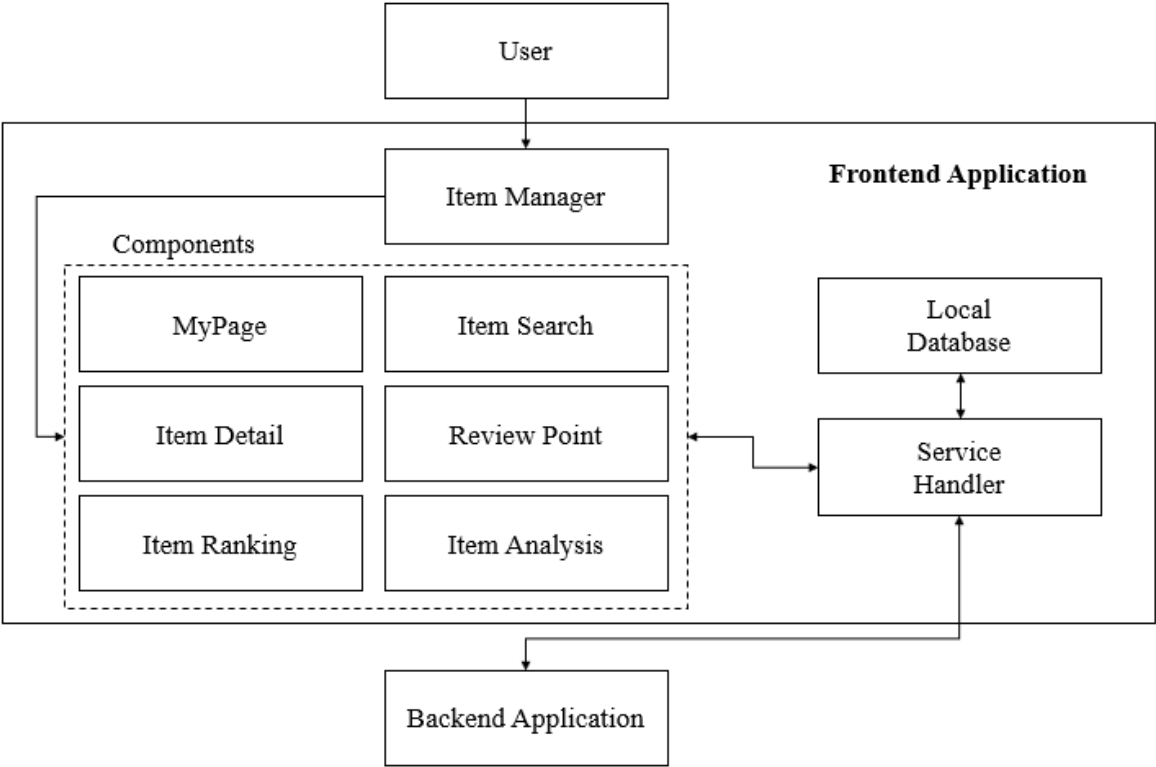


Figure 4. System Architecture - Frontend

Frontend Application is a system that is responsible for interaction with the user. Frontend Application must manage each component via a web framework for mobile environments. The Front Application Item Manager is present in the elements, such as MyPage, Item Search, Item Detail, Point, Item Ranking Item Analysis. The Local Database system manages shared Resource sharing between components. Service Handler performs related work to ensure that each component can communicate with the Backend Application. If each part were to request data needed by Service Handler, the data are transmitted to the Service Handler of Backend Application. At this time, the system requires data with the communication protocol after the conversion.

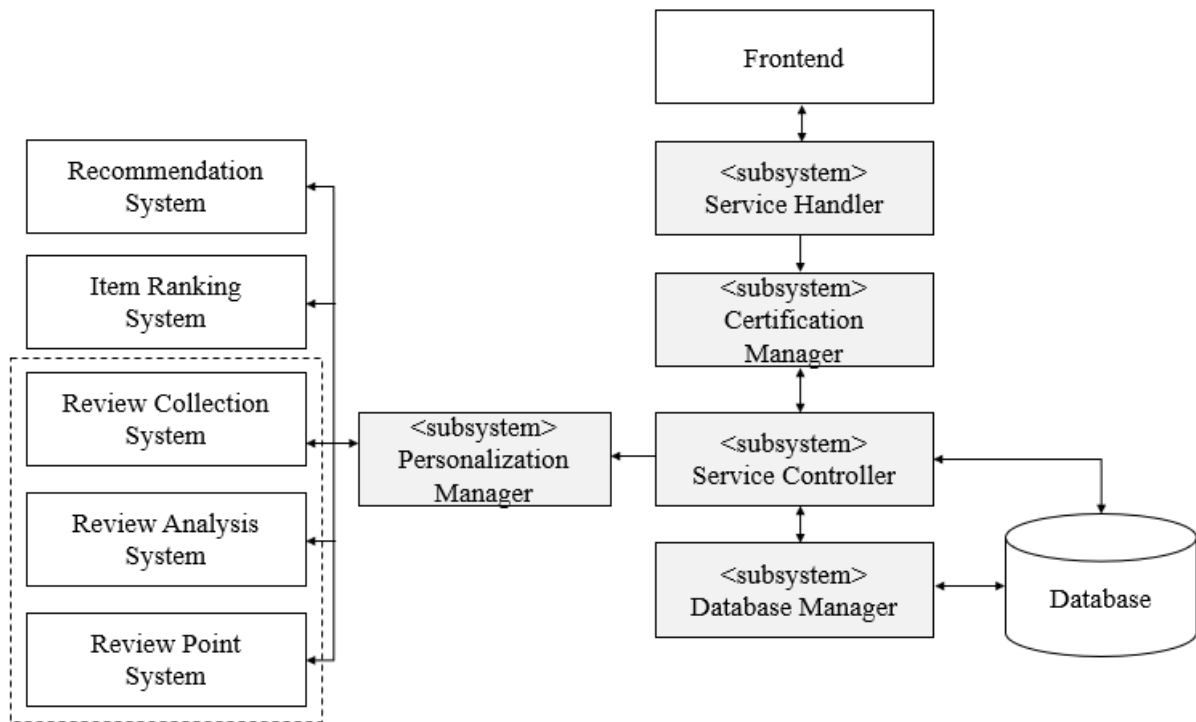


Figure 5. System Architecture - Backend Application

The Backend Application is responsible for performing the tasks requested by the Frontend Application. When Frontend Application is customized requirements from the user forwarded to the Backend Application, the Backend Application should process the job to be actually performed. The first Certification Manager is responsible for the security processing for the authenticated user. If the user is a member of the mall, the Service Controller stores user-related log information to the local database. The Service Controller then asks the user to review the evaluation of the registered Personalization manager. Personalization manager collects, analysis, reliability calculations, and reviews reactivity calculations, each of the operations to the recommendation system, item ranking system, review collection system, review analysis system, review point system to perform such a review compensation amount calculated to review information allocates and manages.

#### 4.5. STEP 6: SKETCH VIEWS AND RECORD DESIGN DECISIONS

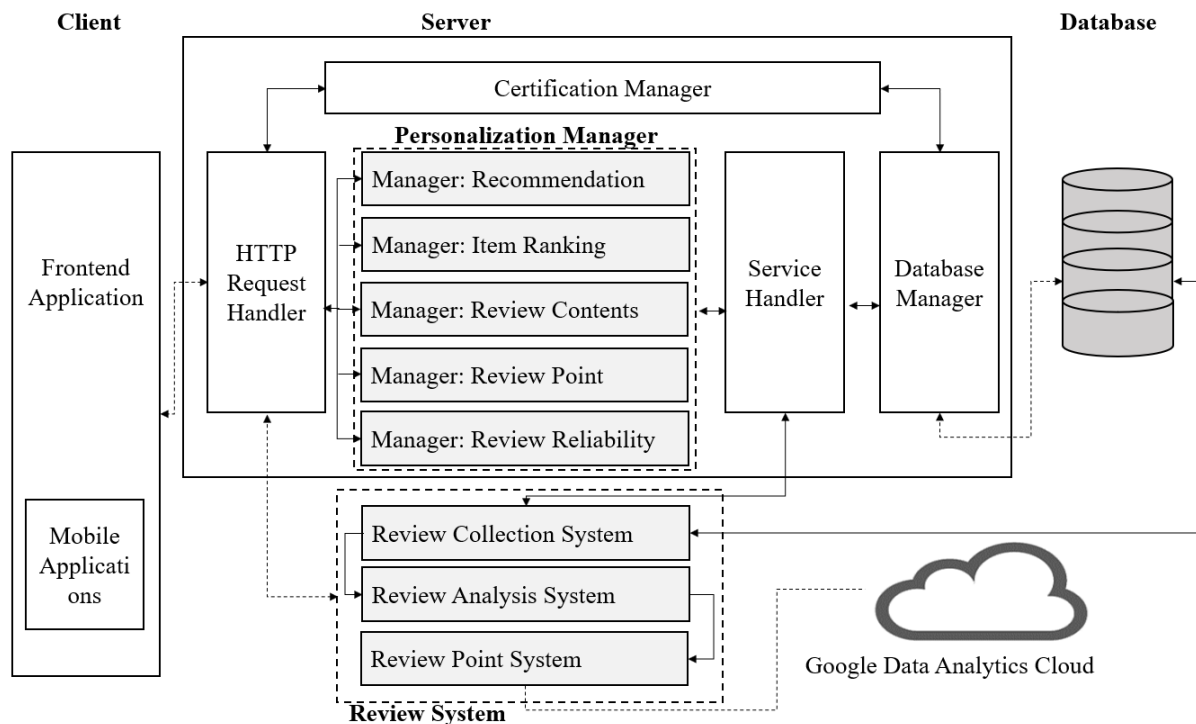


Figure 6. System Architecture - Backend - Overall

The figure above shows the entire operating structure of the Backend System. The HTTP Service Handler is responsible for Frontend Application and direct communication. The Backend System handles and manages all requests for Frontend Application using a Personalization Manager and Review System. Personalization Manager performs such Recommendation, Item ranking, review contents, review point management, review reliability to manage a list of items. And Review System executes Review collection system, review analysis system, review point system, etc. To analyze the review information collected by using the remote Data Analytics Cloud.

The below table summarizes each element's responsibilities

Table 7. Responsibility of the elements for customized recommendation service

Element	Responsibility
<b>Personalization Manager</b>	This element is responsible for storing item ranking, review contents, review point, and review reliability score. Also, this element stores the recommendation related analysis data and the fitting data (e.g., customize fitting service that combine user photos and recommended clothes) after receiving from the review system

<b>Review System</b>	This element collect data from the reviewers in real-time and dispatches it to both collected data and point contents to the data analytics cloud.
<b>Google data analytics cloud</b>	This element is to accelerate time to insights and leave the complexities of data analytics with a comprehensive and serverless data analytics and machine-learning platform. Also, it provides end-to-end data analytics services that surpass conventional limitations on scale, performance, and cost efficiency.

#### 4.6. STEP 7: PERFORM ANALYSIS OF CURRENT DESIGN AND REVIEW ITERATION GOAL AND ACHIEVEMENT OF DESIGN PURPOSE

Not addressed	Partially addressed	Completely addressed	Design decisions made during the iteration
	UC-10		Use reference architecture in order to provide access to real-time data and fast user-responsiveness.
	UC-11		Use reference architecture to provide access to real-time data. No detailed decisions of which secure data transferring technology to use remote service such as cloud-based deep-learning data analytics server (e.g., Google Data Analytics).
	UC-14		Use reference architecture to provide the shopping mall users with customized fitting service. No detailed decisions of which image fitting and data analytics technology to use have been made.

## 5. DESIGN PROCESS: ITERATION 2 (SELECTING TECHNOLOGIES)

In the iteration2, we depict the steps (2 to 7) **to select technologies** in order to enhance the existing recommendation system. Also, it will show a technology tree that helps us choose optimal building blocks when designing the recommendation systems.

Technology choices often influence the system architecture. Therefore, we need to select technologies carefully at the earliest stages of architecture design.

Choosing technologies starts with the identification and selection of technology families that are further instantiated into specific technologies. For example, starting with technology families allows us to make specific technologies interchangeable and thus keep the right level of technology agnosticism to avoid vendor lock-in.

### 5.1. STEP 2: ESTABLISH ITERATION GOAL BY SELECTING DRIVERS

The goal of this iteration is to address CRN-3 (Leverage the team's knowledge about the core technologies (e.g., Cloud, Deep-learning network model, and secure key management) by selecting related technologies to support system requirements in Section 2. In particular, we keep in mind CN-4 (Support 100 simultaneous users by introducing reliable cloud API technologies).

### 5.2. STEP 3: CHOOSE ONE OR MORE ELEMENTS OF THE SYSTEM TO REFINE

The reference architecture selected in the previous iteration (the reference architecture of iteration 1) is decomposed into elements that facilitate the selection of technology families and their related specific technologies

For example, these elements include the Personalization manager (recommendation, item ranking, review content, review point, reviewer reliability), Review system (review collection, review analysis, review point), and data analytics cloud.

### 5.3. STEP 4: CHOOSE ONE OR MORE DESIGN CONCEPTS THAT SATISFY THE SELECTED DRIVERS

The design concepts used in this iteration are externally provided components. Initially, technology families are selected and associated with the elements to be refined. The technology family below represents a group of technologies with common functional purposes.

Especially, Cloud service's fully managed, serverless approach removes operational overhead by handling performance, scalability, availability, and security with cloud-based open APIs. So, we can focus on data analysis instead of managing servers.

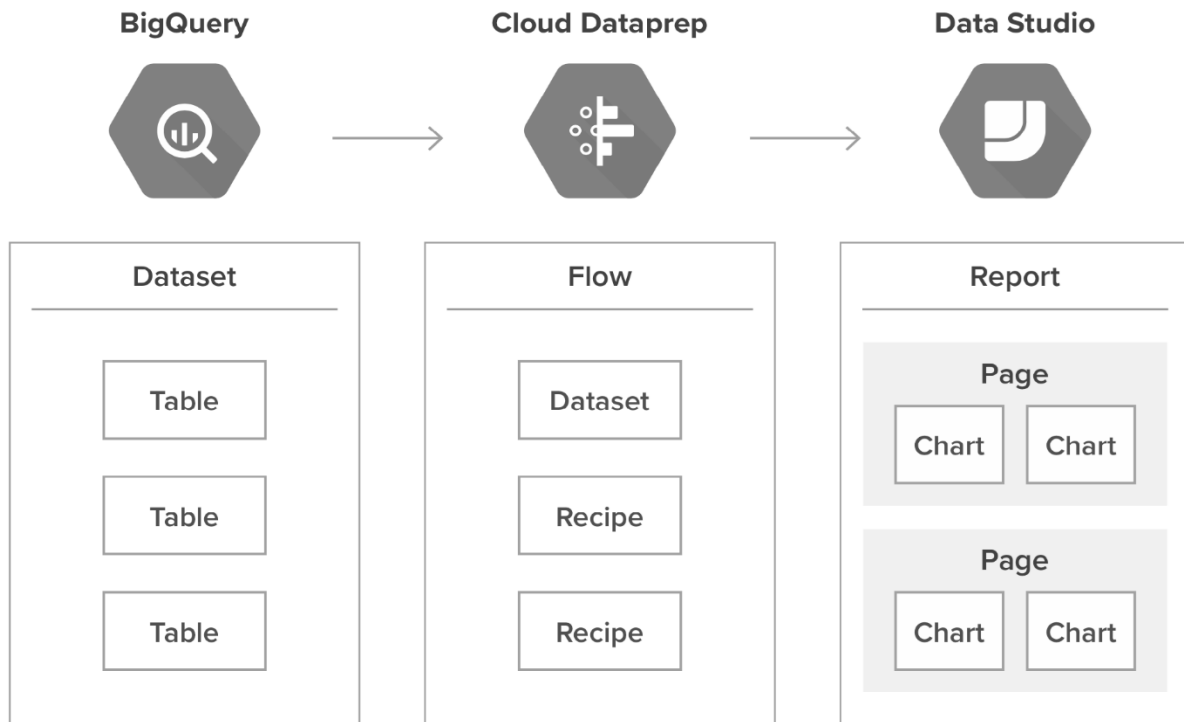


Figure 7. An example of cloud-based data analytics

The below table represent the design concept to analyze review data and recommend suitable items to the users of the personalized shopping mall (e.g., Fitts).

Design decisions and location	Rationale and assumptions
<b>Select the Data analytics family for analyzing review data and recommend suitable clothes.</b>	<p>The recommendation module of the personal manager is a technology family that collects, train, generation, and discrimination for customized fitting service.</p> <p>The goal of the recommendation module of that is to provide fast user-responsiveness as well as accuracy of the fitting service by introducing cloud and external data analytics (e.g., Google Data analytics and cloud).</p>

#### 5.4. STEP 5: INSTANTIATE ARCHITECTURAL ELEMENTS, ALLOCATE RESPONSIBILITIES, AND DEFINE INTERFACES

In this iteration, instantiation is performed by associating specific technologies (e.g., GAN) with the data analytics technology that is previously selected.

Design decision and location	Rationale
<b>Use GAN network model from the review collection module of the review system</b>	As a primary candidate technology, we will select Conditional Analogy Generative Adversarial Network (CAGAN) network

model. It provides the required image matching facility to support (QA-6 fully automated production for deployability)

Alternative	Reason for discarding
<b>SQL DBMS or Key-value store DB</b>	Even though the conventional relational DBMS is quite popular technologies and will satisfy the requirements of the shopping mall data. We have to make a choice and select faster DBMS to give users fast user responsiveness. Key-value store DB supports fast data operations by adopting a hashing algorithm (Optionally based on In-Memory storage).

The below figure represents the design concept of deep-learning model with collected review data to recommend suitable cloths to the users of the Fitts shopping mall. Many image processing problems can be modeled as image-to-image translation problems (e.g. colorizing black-and-white images, translating from a sketch of a scene to a colorful photography, etc.), where an image is fed as input and the output is the modified image. Convolutional neural networks (CNN) are powerful deep learning models that can solve such problems. The more recent model class of Generative Adversarial Networks (GANs) train a model G that learns a data distribution from example data, and a discriminator D that attempts to distinguish generated from training data.

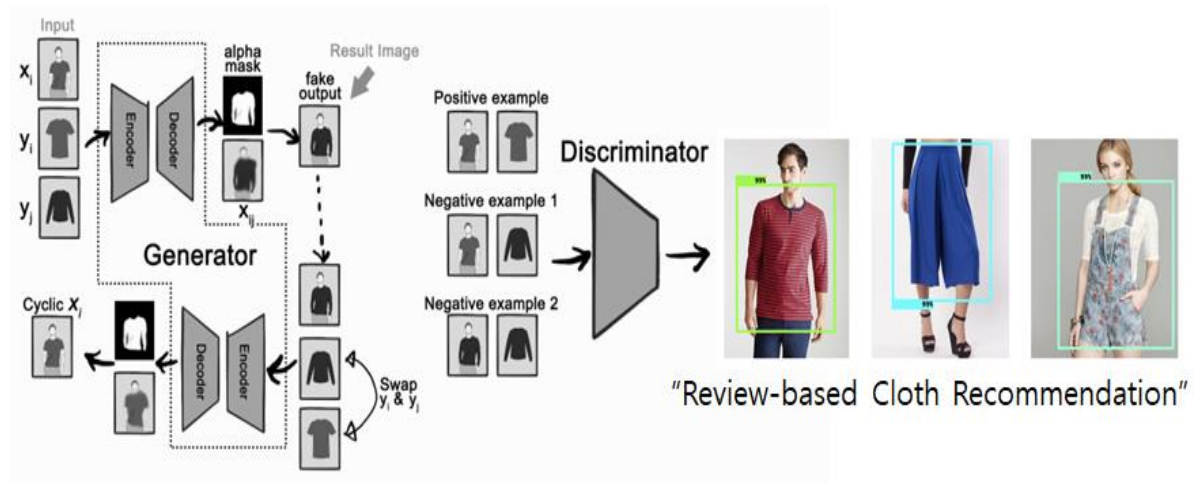


Figure 8. Customized Fitting Service with CAGAN

## 5.5. STEP 6: SKETCH VIEWS AND RECORD DESIGN DECISIONS

In this iteration, instantiation is performed by associating specific technologies with the technology families that were previously selected.



Element	Technology family	Candidate Technology
<b>Personalization Manager</b>	Deep-Learning Framework	CNN, GAN
<b>Review System</b>	Data Processing Framework	MySQL, MongoDB, LevelDB, MemcacheDB
<b>Google data analytics cloud</b>	Data Analytics Engine	Google Cloud Platform Amazon AWS MS Azure

The below figure show the design decision with sketch views as a next step of the iteration 1.

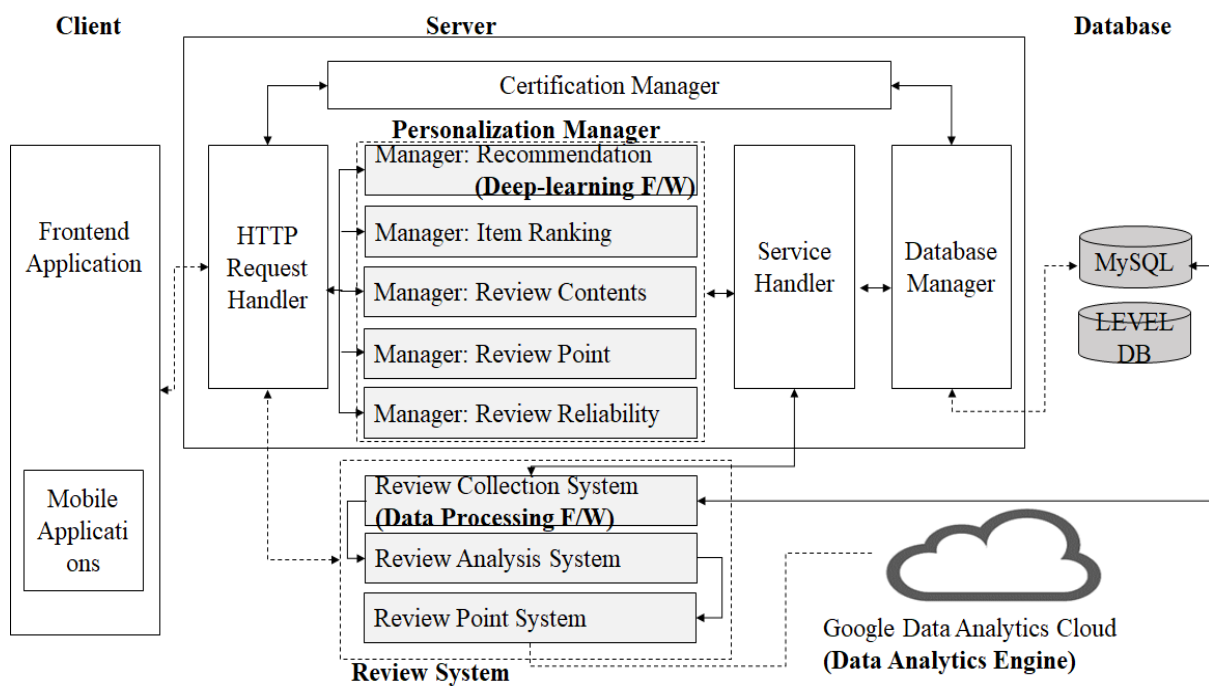



Figure 9. System Architecture - Backend - Overall



The below figure presents the basic and minimum concepts for an end-to-end self-service analytics solution built on the cloud platform (e.g., Google Cloud platform). These concepts are the foundation developers need to understand and iterate on for cloud-based data analytics solution.



Concept	Service	Description
<b>GCP Project</b>	GCP	A project organizes all your GCP resources. You'll need to use an existing project or create a new one.
<b>BigQuery</b>	Dataset	Datasets are top-level containers to organize your tables and views within BigQuery. You need to create at least one dataset before loading data into BigQuery.
<b>BigQuery</b>	Table	A BigQuery table contains individual records organized in rows. Each record is composed of columns (also called fields). A table belongs to a BigQuery Dataset.
<b>Cloud Dataprep</b>	Dataset	A Cloud Dataprep dataset is a reference to a data source such as a BigQuery table or a file in Google Cloud Storage. This is not to be confused with BigQuery tables.
<b>Cloud Dataprep</b>	Recipe	A recipe is a sequence of steps applied to one or multiple Cloud Dataprep datasets to transform the data into the desired output.
<b>Cloud Dataprep</b>	Flow	A flow contains Cloud Dataprep datasets and associated recipes to define a logical order for transforming the data. A flow can be scheduled to prepare data on a regular basis.
<b>Data Studio</b>	Report	A report with Data Studio contains various pages, each combining charts (tables, bars, pies, etc.) to create interactive dashboards.
<b>Data Studio</b>	Dimension	A dimension is a particular field within a table that describes the characteristics of the data (e.g. product name, region, or customer).
<b>Data Studio</b>	Metric	A metric is a numeric value that quantifies something relative to a dimension. For example, a quantity or a discount are metrics relative to a product name. Metrics are often expressed with an aggregation function (e.g. sum of quantities, max price, or average discount).

Figure 10. Cloud Platform for API-based data analytics

The next-gen shopper expects to walk into an experience, not a store. An experience that combines AI, Cloud, and mobile technologies. This creates intuitive, relevant and engaging experiences – like the interactive fitting room.

Once, the Deep Learning technology infers a customer's suitable clothes with collected review data and customer's photos. AI technologies match in-store merchandise with the brands and styles the person likes on their social media channels. A deep learning-based fitting recommendation system displays the execution results. This allows the customers to select suitable items they want to wear in real. They can pay the money by using their mobile payment device or the Fitts points.



Figure 11. Personalized Fashion Recommendation with Deep-Learning

## 5.6. STEP 7: PERFORM ANALYSIS OF CURRENT DESIGN AND REVIEW ITERATION GOAL AND ACHIEVEMENT OF DESIGN PURPOSE

Not addressed	Partially addressed	Completely addressed	Design decisions made during the iteration
	UC-7		Use Google Data Analytics and GAN network model to display customized fitting service with reliable security and fast image fitting operations when users see the items in their cart before deciding purchase it finally.
	UC-11		Use In-memory DBMS for getting the predicted user responsiveness even though 100 users simultaneously comment review data and display review-based customized fitting service with deep-learning such as GAN
	UC-14		Use additional database server for security related database (e.g., coupon, point) from the existing database servers to avoid unintended data access and handle secure data.

## 6. DESIGN PROCESS: ITERATION 3 (QUALITY ATTRIBUTES SCENARIOS DRIVERS)

In iteration 3, we focus on the Quality Attributes Scenarios driver. The need to review the driver is because we decided to integrate a recommendation system with the existing Fitts system architecture.

### 6.1. STEP 2: ESTABLISH AN ITERATION GOAL BY SELECTING DRIVERS

The driver related to iteration 3 is the Quality Attribute Scenarios such as below:

ID	Quality attribute	Scenario	Associated use-case
QA-1	Performance	The app view refresh time when receiving the user's request must be less than 1 second.	UC-4, UC-7
QA-2	Performance	Time for loading the data should be less than 1 second.	UC-3, UC-4, UC-7
QA-3	Security	Because of the point system working as the real money system, so it needs the security level like the banking system.	UC-8, UC-10, UC-14
QA-4	Scalability	System must be large enough to adapt to the huge amount of user data every day, mostly for image data which costs a lot of space.	UC-1, UC-5, UC-6, UC-9
QA-5	Availability	The system shall continue to operate with no downtime if any single node or component fails	All use cases
QA-6	Deployability	The system deployment procedure shall be fully automated and support a number of environments: development, test, and production	All use cases

## 6.2. STEP 3: CHOOSE ONE OR MORE ELEMENTS OF THE SYSTEM TO REFINE

The existing system uses conventional client-server database system. The recommend system will involve the learning of user data such as stated in iteration 2 (**select technologies**). Here we will not refine any system, but we will examine the entire system whether it is still relevant with the current Quality Attribute Scenarios.

## 6.3. STEP 4: CHOOSE ONE OR MORE DESIGN CONCEPTS THAT SATISFY THE SELECTED DRIVERS

The design concept for this iteration will follow the one stated in iteration 1 and 2.

## 6.4. STEP 5: INSTANTIATE ARCHITECTURAL ELEMENTS, ALLOCATE RESPONSIBILITIES, AND DEFINE INTERFACES

Design Decisions and Location	Rationale and Assumptions
<b>The scenario for QA-4 will be updated.</b> <b>Changes to Q-4:</b> <b>“System must be large enough to adapt to the huge amount of user data every day, mostly for data learning for recommendation system.”</b>	The scenario is not relevant to the proposed system.
<b>New QA-7 will be added.</b> <b>Quality Attribute: Reliability</b> <b>Scenario: The real-time data streaming for learning of data must be reliable between user device and Fitts’s server.</b> <b>Associated use-case: UC-4 and UC-7</b>	The new scenario is needed for the proposed system.

## 6.5. STEP 6: SKETCH VIEWS AND RECORD DESIGN DECISIONS

The new Quality Attribute Scenario driver such as below:

ID	Quality attribute	Scenario	Associated use-case
<b>QA-1</b>	Performance	The app view refresh time when receiving the user's request must be less than 1 second.	UC-4, UC-7
<b>QA-2</b>	Performance	Time for loading the data should be less than 1 second.	UC-3, UC-4, UC-7
<b>QA-3</b>	Security	Because of the point system working as the real money system, so it needs the security level like the banking system.	UC-8, UC-10, UC-14
<b>QA-4</b>	Scalability	System must be large enough to adapt to the huge amount of user data every day, mostly for data learning for recommendation system.	UC-1, UC-5, UC-6, UC-9
<b>QA-5</b>	Availability	The system shall continue to operate with no downtime if any single node or component fails	All use cases
<b>QA-6</b>	Deployability	The system deployment procedure shall be fully automated and support a number of environments: development, test, and production	All use cases
<b>QA-7</b>	Reliability	The real-time data streaming for learning of data must be reliable between user device and Fitts's server	UC-4, UC-7

## 6.6. STEP 7: PERFORM ANALYSIS OF CURRENT DESIGN AND REVIEW ITERATION GOAL AND ACHIEVEMENT OF DESIGN PURPOSE

Not addressed	Partially addressed	Completely addressed	Design decisions made during the iteration
		QA-4	Changes in the scenario of the quality attribute
		QA-7	The new Quality Attribute Scenario is added because we need to address about the reliability when adding the proposed recommendation system to our existing system architecture.

## 7. DESIGN PROCESS: ITERATION 4 (CONSTRAINTS DRIVERS)

The drivers for this iteration are regarding to the constraint driver. Constrains driver contains of multiple constrains that pre-defined to ensure the quality of the system during operating.

### 7.1. STEP 2: ESTABLISH ITERATION GOAL BY SELECTING DRIVERS

This iteration step we focus on the constrains that provide reliable usable of the Fitts platform.

ID	Constraint
<b>CON-7</b>	Fitts system just can access to use Users data if and only when Users allow to provide the data.
<b>CON-8</b>	Must use Users data for only training for recommend system purpose
<b>CON-2</b>	Just recommend system is allowed to access user's Fitts database

### 7.2. STEP 3: CHOOSE ONE OR MORE ELEMENTS OF THE SYSTEM TO REFINE

Recommend system is actually just the Deep Learning system, which was prove by many articles that needs a huge amount of data to operate. Because of that, we need to add more constrains to existing system to make it more reliable and transparent to user in the term of using user's data.

### 7.3. STEP 4: CHOOSE ONE OR MORE DESIGN CONCEPTS THAT SATISFY THE SELECTED DRIVERS

In this iteration the only selection design decision involves choosing tactics to satisfy the performance, extensibility, performance and usability quality attributes.

### 7.4. STEP 5 AND 6: INSTANTIATE ARCHITECTURAL ELEMENTS, ALLOCATE RESPONSIBILITIES, DEFINE INTERFACES AND SKETCH VIEWS AND RECORD DESIGN DECISIONS

In this iteration most of the decisions are about instantiation. The only selection decision involves choosing tactics to satisfy the availability and performance quality attributes.



Design Decisions and Location	Rationale and Assumptions
<b>Use Pytorch as a Deep Learning framework for Recommend System.</b>	Pytorch is one of biggest Deep Learning Framework in the industry now. It is backed by Facebook – the biggest Social Network Company in the world. Pytorch provide easy way to learn and implement into the existing system, but also provide the powerful access to utilize the hardware resources.

## 7.5. STEP 7: PERFORM ANALYSIS OF CURRENT DESIGN AND REVIEW ITERATION GOAL AND ACHIEVEMENT OF DESIGN PURPOSE

Not addressed	Partially addressed	Completely addressed	Design decisions made during the iteration
		CON-2	Added approval to access database for only recommendation system
	UC-10		Use reference architecture in order to provide access to real-time data and fast user-responsiveness.
	UC-11		Use reference architecture to provide access to real-time data. No detailed decisions of which secure data transferring technology to use remote service such as cloud-based deep-learning data analytics server (e.g., Google Data Analytics).

## 8. DESIGN PROCESS: ITERATION 5 (ARCHITECTURAL CONCERN DRIVERS)

The drivers for this iteration are regarding to **the architectural concern driver**. Architectural concerns driver contains of multiple concerns that pre-defined to ensure the performance of the development team.

### 8.1. STEP 2: ESTABLISH ITERATION GOAL BY SELECTING DRIVERS

In this iteration step, we focus on the concerns that provide reliable usable of the Fitts platform.

ID	Concern
CRN-1	Must establish an understanding about the overall system structure
CRN-3	Leverage the team's knowledge about core technologies using in the system
CRN-4	Rewrite the code of conduct for development team and privacy term for users

We want to address and redefine those above concerns in the way it will support for the new system.

### 8.2. STEP 3: CHOOSE ONE OR MORE ELEMENTS OF THE SYSTEM TO REFINE

We want to refine the documents as well as the non-functional requirements for the recommend system.

### 8.3. STEP 4: CHOOSE ONE OR MORE DESIGN CONCEPTS THAT SATISFY THE SELECTED DRIVERS

In this iteration the only selection design decision involves choosing tactics to satisfy the performance, extensibility, performance and usability quality attributes; and the external component to support Deep Learning framework.

Design Decisions and Location	Rationale and Assumptions
<b>Introduce active redundancy by replicating the application server and other critical components such as the database</b>	By replicating the critical components, the system can withstand the failure of one of the replicated elements without affecting functionality.

**Introduce to prevent the exception by provide more security protocol when working with Users data**

By using more security protocol when working with Users data, the system can prevent the exception of leakage on data.

#### 8.4. STEP 5 AND 6: INSTANTIATE ARCHITECTURAL ELEMENTS, ALLOCATE RESPONSIBILITIES, DEFINE INTERFACES AND SKETCH VIEWS AND RECORD DESIGN DECISIONS

Most of the work in this iteration is just aim for the document and working code of conduct, it is not impact to the overall system structure.

#### 8.5. STEP 7: PERFORM ANALYSIS OF CURRENT DESIGN AND REVIEW ITERATION GOAL AND ACHIEVEMENT OF DESIGN PURPOSE

Not addressed	Partially addressed	Completely addressed	Design decisions made during the iteration
		CRN-1	The system structure now including the recommend system as well, it requires more in-depth knowledge on how those 2 systems working with each other.
		CRN-3	Development team is required to understand more knowledge on the Artificial Intelligent field as well as Deep Learning aspect. Moreover, the team must understand about the micro services pattern, which relevant to the new architecture.
	CRN-4		Rewrite the code of conduct for development team and privacy term for users

## 9. CONCLUSION

ADD is a powerful method for architecture design that distinguishes itself from other design methods because it focuses on system decomposition from a quality attributes' perspective. ADD can be used in conjunction with other SEI methods such as the Quality Attribute Workshop for gathering requirements for input to ADD or the Architecture Tradeoff Analysis Method® (ATAM®) to evaluate architectures that result from applying ADD. ADD can also be used within or adapted to most any development life cycle or process (e.g., evolutionary, waterfall, spiral, Rational Unified Process [RUP], or agile development).

This document describes a novel system architecture of Fitts designed using the ADD method in order to improve the naive recommendation system of existing Fitts systems. We address a state-of-the-art clothing recommendation system based on artificial intelligence and review data of the users. It gives customers a preview of what a customer would look like before they buy it at an internet shopping mall. The proposed recommendation system is improved by deep learning-based fitting service by combining user photos and appropriate clothes. The proposed system architecture conforms to ADD iteration steps to satisfy system requirements, user case scenarios, architectural concerns, and design constraints. Even though the architecture driver is challenging, we performed carefully ADD Iteration as long as the primary design structure of the recommended system is maintained.

This document consists of five iterations as follows.

The iteration1 describes the design of a greenfield system. Therefore, the goal of the 1st iteration is to establish an initial overall structure of the system. Although the 1st iteration is driven by a general architectural concern, the architect has to keep in mind all of the drivers. Primarily, the architect handles constraints and quality attributes.

In the iteration2, we depict the steps (2 to 7) to select technologies in order to enhance the existing recommendation system. Also, it shows a technology tree that helps us choose optimal building blocks when designing the recommendation systems. Technology choices often influence the system architecture. Therefore, we need to select technologies carefully at the earliest stages of architecture design.

In iteration 3, we focus on the Quality Attributes Scenarios driver. The need to review the driver is because we decided to integrate a recommendation system with the existing Fitts system architecture.

The drivers for the 4th iteration is regarding the constraint driver. The constraint driver contains multiple constraints that pre-defined to ensure the quality of the system during operating. In this iteration step, we focus on the constraints that provide reliably usable of the Fitts platform.

The drivers for the 5th iteration is regarding the architectural concern driver. Architectural concerns driver contains multiple concerns that pre-defined to ensure the performance of the development team. In this iteration step, we focus on the concerns that provide reliably usable of the Fitts platform. In this iteration, the only selected design decision involves tactics to satisfy the performance, extensibility, performance, and usability quality attributes; and the external component to support the Deep Learning framework.



## APPENDIX. TERMINOLOGY

- UC: User Case
- QA: Quality Attribute
- CN: Constraints
- CRN: Architectural Concerns
- TBD: To-Be-Defined
- TBA: To-Be-Added

## APPENDIX. KANBAN BOARD

- [https://github.com/skkuse-adv/2019Fall\\_team2/projects/1?fullscreen=true](https://github.com/skkuse-adv/2019Fall_team2/projects/1?fullscreen=true)

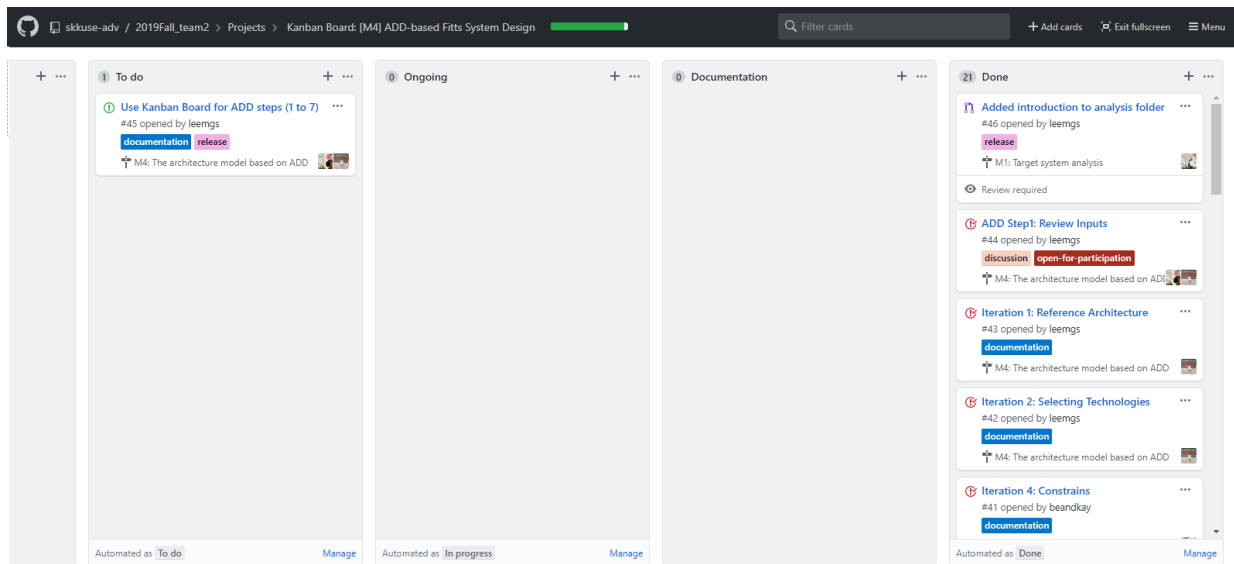


Figure 12. Kanban Board for ADD iteration.

## APPENDIX. GITHUB REPOSITORY

We archive the JAVA source code files that are decompiled by reverse-engineer approaches. All data are available at the GitHub Repository as following:

- [https://github.com/skkuse-adv/2019Fall\\_team2](https://github.com/skkuse-adv/2019Fall_team2)

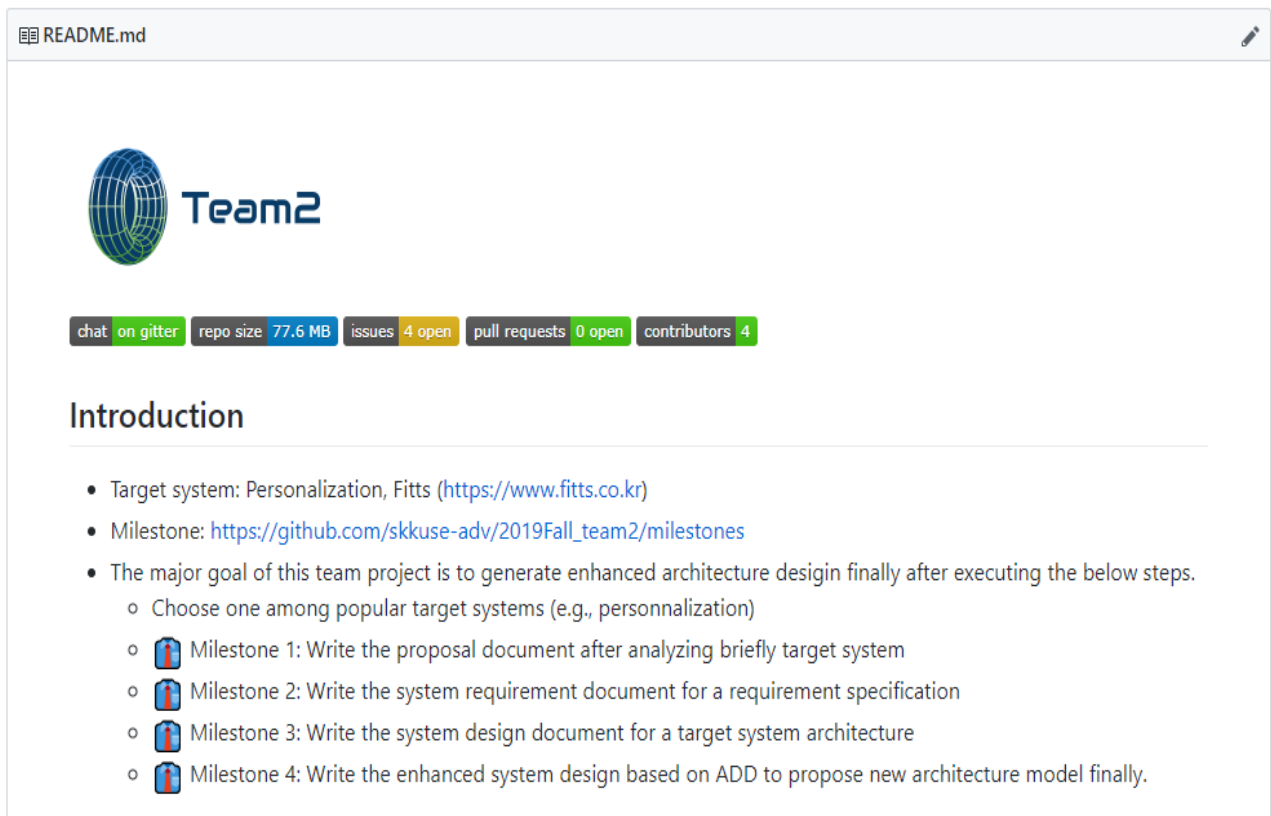


Figure 13. GitHub Repository of Team2

## APPENDIX. REFERENCES

1. Fitts, SNS & Review based On-line Shopping Mall, [Online]. Available: <https://fitts.co.kr/>
2. **ADD 3.0: Rethinking Drivers and Decisions in the Design Process**, Apr-2015, [Online]. Available: <https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=436536>
3. Wojcik, Rob, Felix Bachmann, Len Bass, Paul Clements, Paulo Merson, Robert Nord, and Bill Wood. **Attribute-driven design (ADD), version 2.0**. No. CMU/SEI-2006-TR-023. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2006.
4. Jeon, Sanghoon, Myungjin Han, Eunseok Lee, and Keun Lee. "**Quality attribute driven agile development**." In 2011 Ninth International Conference on Software Engineering Research, Management and Applications, pp. 203-210. IEEE, 2011.
5. Nord, Robert L., and James E. Tomayko. "**Software architecture-centric methods and agile development**." *IEEE software* 23, no. 2 (2006): 47-53.
6. Bachmann, Felix, and Len Bass. "**Introduction to the attribute driven design method**." In Proceedings of the 23rd international conference on Software engineering, pp. 745-746. IEEE Computer Society, 2001.
7. Bass, Len, Mark Klein, and Felix Bachmann. "**Quality attribute design primitives and the attribute driven design method**." In International Workshop on Software Product-Family Engineering, pp. 169-186. Springer, Berlin, Heidelberg, 2001.
8. Bajpayee, Payel, and Hassan Reza. "**Toward Quality Attribute Driven Approach to Software Architectural Design**." *Journal of Software Engineering and Applications* 10, no. 6 (2017): 483.
9. Gysel, Michael, Lukas Kölbener, Wolfgang Giersche, and Olaf Zimmermann. "**Service cutter: A systematic approach to service decomposition**." In *European Conference on Service-Oriented and Cloud Computing*, pp. 185-200. Springer, Cham, 2016.
10. Kuo, Tsai-Chi, Shana Smith, Gregory C. Smith, and Samuel H. Huang. "**A predictive product attribute driven eco-design process using depth-first search**." *Journal of cleaner production* 112 (2016): 3201-3210.
11. Bachmann, Felix, and Len Bass. "**Introduction to the attribute driven design method**." In *Proceedings of the 23rd international conference on Software engineering*, pp. 745-746. IEEE Computer Society, 2001.
12. Bass, Len, Mark Klein, and Felix Bachmann. "**Quality attribute design primitives and the attribute driven design method**." In *International Workshop on Software Product-Family Engineering*, pp. 169-186. Springer, Berlin, Heidelberg, 2001.