Deliverable #2 Template

SE 3A04: Software Design II – Large System Design

Tutorial Number: T03
Group Number: G06
Group Members:

- Virochaan Ravichandran Gowri
- Alex Yoon
- Noah Goldschmied
- Krish Dogra
- Leo Vugert

IMPORTANT NOTES

- Please document any non-standard notations that you may have used
  - *Rule of Thumb*: if you feel there is any doubt surrounding the meaning of your notations, document them
- Some diagrams may be difficult to fit into one page
  - Ensure that the text is readable when printed, or when viewed at 100% on a regular laptop-sized screen.
  - If you need to break a diagram onto multiple pages, please adopt a system of doing so and thoroughly explain how it can be reconnected from one page to the next; if you are unsure about this, please ask about it
- Please submit the latest version of Deliverable 1 with Deliverable 2
  - Indicate any changes you made.
- If you do NOT have a Division of Labour sheet, your deliverable will NOT be marked

1 Introduction

This section should provide an brief overview of the entire document.

1.1 Purpose

State the purpose and intended audience for the document.

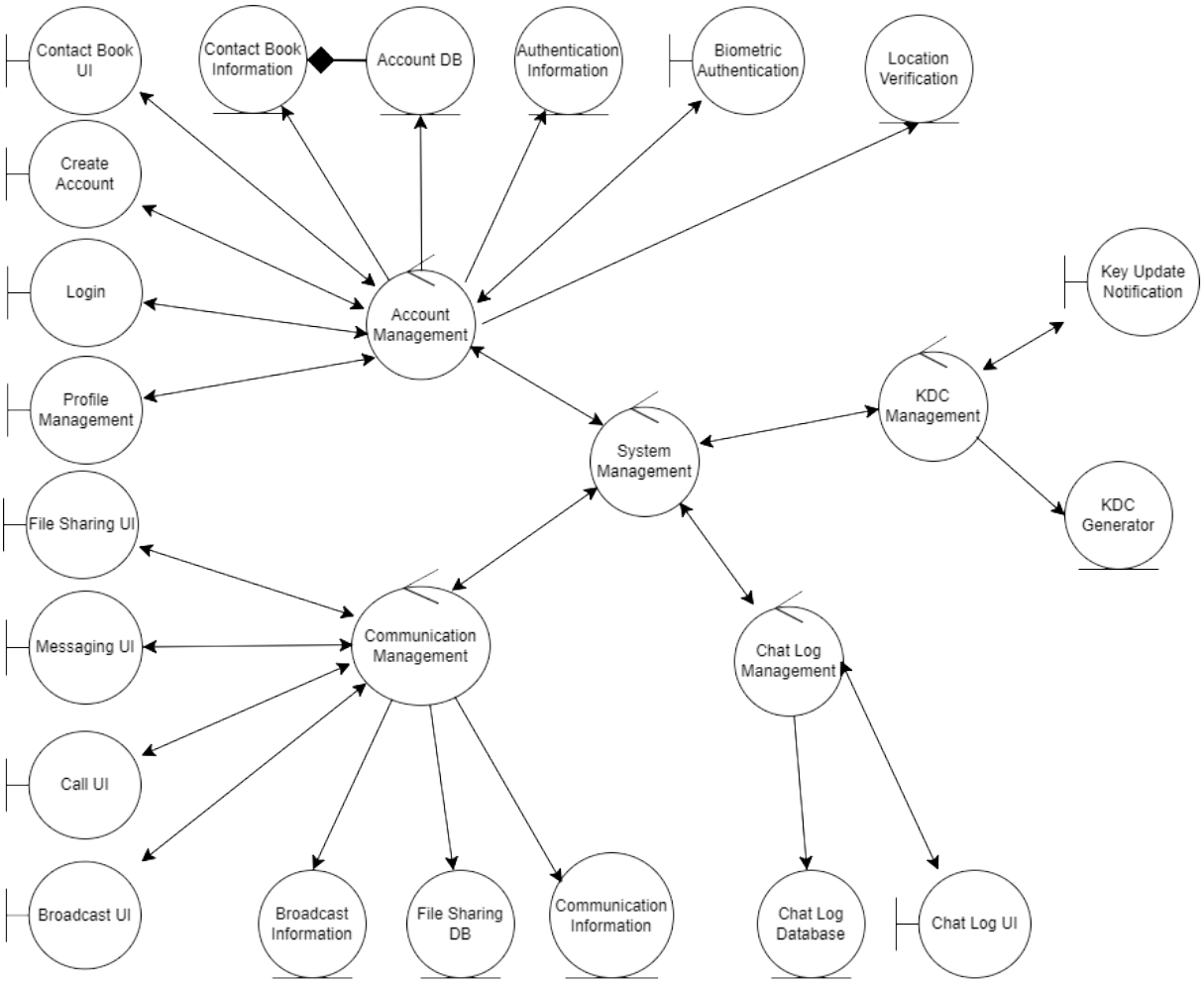
1.2 System Description

Give a brief description of the system. This could be a paragraph or two to give some context to this document.

1.3 Overview

Describe what the rest of the document contains and explain how the document is organised (e.g. "In Section 2 we discuss...in Section 3...").

2 Analysis Class Diagram



3 Architectural Design

This section should provide an overview of the overall architectural design of your application. Your overall architecture should show the division of the system into subsystems with high cohesion and low coupling.

3.1 System Architecture

- Identify and explain the overall architecture of your system
- Be sure to clearly state the name of the architecture you used (this is the name of the architectural pattern, not the name of your system)
- Provide the reasoning and justification of the choice of architecture
- Provide a structural architecture diagram showing the relationship among the subsystems (if appropriate)
- List any design alternatives you considered, but eliminated (and explain why you eliminated them)

There a 3 primary subsections within the system that utilize their own distinct architecture style but still exist within the overall larger architecture style:

Subsystem	Purpose	Architectural Style
Account Management	Create, Login and Manage Account Information and Functions	Repository
Communication Management	Provides communication functionality between different agents and users.	Repository
KDC Management	Provides encryption and decryption functionalities	Pipe and Filter

The subsystems employ the Repository and the Pipe and Filter architecture styles and the relationship and functionality are further defined in section 3.2.

We also include 4 databases on the model level, an account database, a contacts database, a KDC database and a Chat Log database.

-Talk about MVC Here

We chose the Repository Architecture Style for the Account Management and Communication Management subsystem since it supports easy scalability and utilizes a passive data store meaning the agents are active in controlling the data flow logic. This enables different agents to access different aspects of the system and different parts of data at the same time which will be beneficial for our system. Also having easy scalability through the introduction of new agents is crucial for our use case and easily implemented through this architecture style.

The Pipe and Filter Architecture Style was chosen for the KDC since it provides efficient processing of data with a high throughput and flexibility. Since we are expecting many messages at the same time throughput is a crucial factor to consider and the pipe and filter architecture allows for easy scaling of throughput through concurrency. It also provides simplicity by sectioning different parts of the KDC into filters for encryption and decryption. Finally it also allows for reusability and room to expand by simplify modifying or adding more pipes and filters.

3.2 Subsystems

Provide a list of your subsystems, with a brief description of each. Be sure to document its purpose and relationship to other subsystems.

We have 3 primary subsystems that are contained within the overall system which provide distinct functionalities and which also have their own distinct architectural styles for implementation. Our Three distinct subsytems include: Account Management, Communication Management Service, and KDC Managment.

The Account Management subsystem provides users with ability to create and login to their account and is responsible for ensuring that only authorized users account access the system by interfacing with the external apis. It also manages users contacts and interfaces with the communication management service to allow for communication and the authentication management for authentication management. It utilizes the repository architecture style.

The Communication Management Service provides users with the ability to contact other authorized user by sending and recieving messages. It also allows for the sending and recieving of files through file transfer, the reporting of messages and recieving of announcement board posts. It interacts with the authentication managment to ensure the authentication of users, the KDC Management for encryption and decryption of messages and the chat log managment and adheres to the Repository architecture style.

The KDC Management provides the encryption and decryption functions for the system. It interacts with the communication module and utilizes a pipe and filter architecture style.

4 Class Responsibility Collaboration (CRC) Cards

Class Name: System Management (Controller)	
Responsibility:	**Collaborators:**
Knows Account Management	Account Management
Knows Communication Management	Communication Management
Knows Chat Log Management	Chat Log Management
Knows KDC Management	KDC Management

Class Name: Communication Management (Controller)	
Responsibility:	**Collaborators:**
Knows System Management	System Management
Knows File Sharing UI	File Sharing UI
Knows Call UI	Call UI
Knows Broadcast UI	Broadcast UI
Knows Broadcast Information	Broadcast Information
Knows File Sharing DB	File Sharing DB
Knows Communication Information	Communication Information
Knows Communication DB	Communication DB

Class Name: File Sharing UI (Boundary)	
Responsibility:	**Collaborators:**
Knows Communication Management	Communication Management
Handles the User Interface for file sharing	
Handles the encryption and decryption of files	

Class Name: Messaging UI (Boundary)	
Responsibility:	**Collaborators:**
Knows Communication Management Handles the User Interface for messaging Handles the encryption and decryption of messages	Communication Management

Class Name: Call UI (Boundary)	
Responsibility:	**Collaborators:**
Knows Communication Management Handles the User Interface for calls Handles the encryption and decryption of calls	Communication Management

A Division of Labour

Include a Division of Labour sheet which indicates the contributions of each team member. This sheet must be signed by all team members.

Class Name: Broadcast UI (Boundary)	
Responsibility:	**Collaborators:**
Knows Communication Management Handles the User Interface for broadcasts Handles the encryption and decryption of broadcasts	Communication Management

Class Name: Broadcast Information (Entity)	
Responsibility:	**Collaborators:**
Knows Communication Management Knows what users belong to broadcasts Knows which users can message in broadcasts	Communication Management

Class Name: File Sharing DB (Entity)	
Responsibility:	**Collaborators:**
Knows Communication Management Knows what users sent which files Knows which users can view individual files Knows where all files that have been sent are stored	Communication Management

Class Name: Communication Information (Entity)	
Responsibility:	**Collaborators:**
Knows Communication Management Knows what users belong to which chat Knows which users can message in which chat	Communication Management

Class Name: Chat Log Management (Controller)	
Responsibility:	**Collaborators:**
Knows System Management Knows Chat Log Database Knows Chat Log UI	System Management Chat Log Management Chat Log UI

Class Name: Chat Log Database (Entity)	
Responsibility:	**Collaborators:**
Knows Chat Log Management Knows what user sent each message Knows the identifiers of users and the date, time, and content of message	Chat Log Management

Class Name: Chat Log UI (Boundary)	
Responsibility:	**Collaborators:**
Knows Chat Log Management Handles presentation of Chat Logs	Chat Log Management