Project Name: Storage Admin System

Team Members
Mai Lor
Thanh Tran
Greg Payne

Professor: Ismail Bile Hassan

Last updated: 12/8/2022

Table of Contents

XVI. State Diagrams
    A. Account Creation
    B. Data Storage

XVII. Updated Use Case Diagram With Include and Extend

Vision

A more affordable and scalable alternative to on-premise hard drives or storage networks is cloud storage. A limited amount of data may be stored on computer hard disks. Users must move files to an external storage device when their internal storage is full.

We can use cloud storage to store data and files in an offsite location that you can access over either a dedicated private network connection or the open internet. A third party cloud provider will be in charge of any data you move off-site for storage. The supplier guarantees that you always have access to the data by hosting, securing, managing, and maintaining the servers and related infrastructure.

Scope

Cloud storage employs servers to store data, similar to on-premise storage networks, except the data is delivered to servers that are located off-site. The majority of servers you use are virtual machines that are housed on actual servers. In order to keep up with demand, the supplier builds more virtual servers as your storage requirements grow.

Business Case

Direct-attached storage (DAS) is a challenge, and storage networks can provide answers. It describes the experiences of IT decision-makers and implementers who have installed SAN (Storage Area Network)solutions to deal with the significant issues their firms are currently facing, which are being overtaken by pricey, ineffective, and challenging-to-manage DAS solutions. The Business Case for Storage Networks discusses issues including storage expansion and growing consumption, the IT department's function as a cost center, and how SAN technologies may help you save money over the long term so that you can make an educated choice about your storage networking investment.

Feasibility Studies

Our memory storage system is feasible because memory storage is in high demand for clients with large amounts of data. Many corporations outsource memory storage because it is more cost-effective than buying their own storage servers and devices. Our storage system is easily integrated with the networks of our clientele, and once we grant access, they can easily access their data at any time. We have operational feasibility because our choice to focus on memory storage gives us a specialized focus or niche.

Software development Methodology

To develop the software required for our memory storage system, we will use the agile method of software development. We will complete efficient, iterative steps in phases with the following primary focus areas:

1. Clearly define our users' needs before creating any software
2. Design portions of the memory storage software in focused bursts called sprints. One such sprint will be to design our network topology.
3. Program for upkeep of our system. This step will include defragmenting our storage servers, software updates and preparing a sophisticated plan to prevent cyberattacks.
4. Vigilant auditing and adjusting our design. We will catch system weaknesses early by logging what packet losses, system uptime, customer satisfaction, and more.
5. Incorporate a philosophy of steady refinement. We will continue to add sprints after the system is live to meet new challenges that appear.

Proposed System Functionalities

The primary functionality of our system is to store memory in a way that is organized, efficient and adaptable. Our schematic is divided according to the type of data to be stored, by file size and by the speed and frequency of access. Usernames and passwords, for example, will be available instantly, while data that is used less often, like educational materials and source code will be stored in slower, more affordable formats. As mentioned in the previous slide, our system functionalities will improve over time by intermittently implementing sprints to meet our customers' needs.

Hardware and Software Requirements

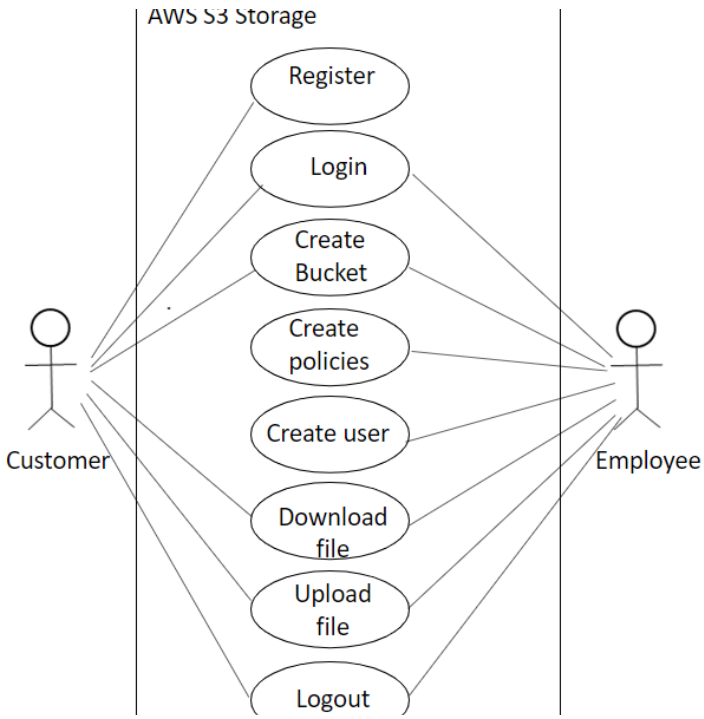
The software being used will be the AWS Storage Gateway. The AWS Storage Gateway is a service provided by Amazon that allows users to integrate existing on-premise applications to their storage service, S3.

- 16GiB of reserved RAM for file Gateway
- 80 GiB of disk space
- Users are required to have internet access, DNS services,and IP address

Use Case

Actors	Use case/Functionalities
Employee	- Login - Create Bucket - Create policies - Create user - Download files - Upload files - Logout
Customer	- Register - Login - Create Bucket - Download files - Upload files - Logout

Use Case Diagrams



Software Architecture

The software design that we came up with is using the Amazon Web Service (AWS) S3 Bucket for storage space. AWS S3 Bucket is a pay per use model where you only pay what you use. A Bucket is defined as a “container” for objects such as photos, videos, documents, etc… This service is considered to be easy to use and is scalable. Although this is a pretty great service, there can be some major risk factors. This document will explain the risk factors and how to avoid them.

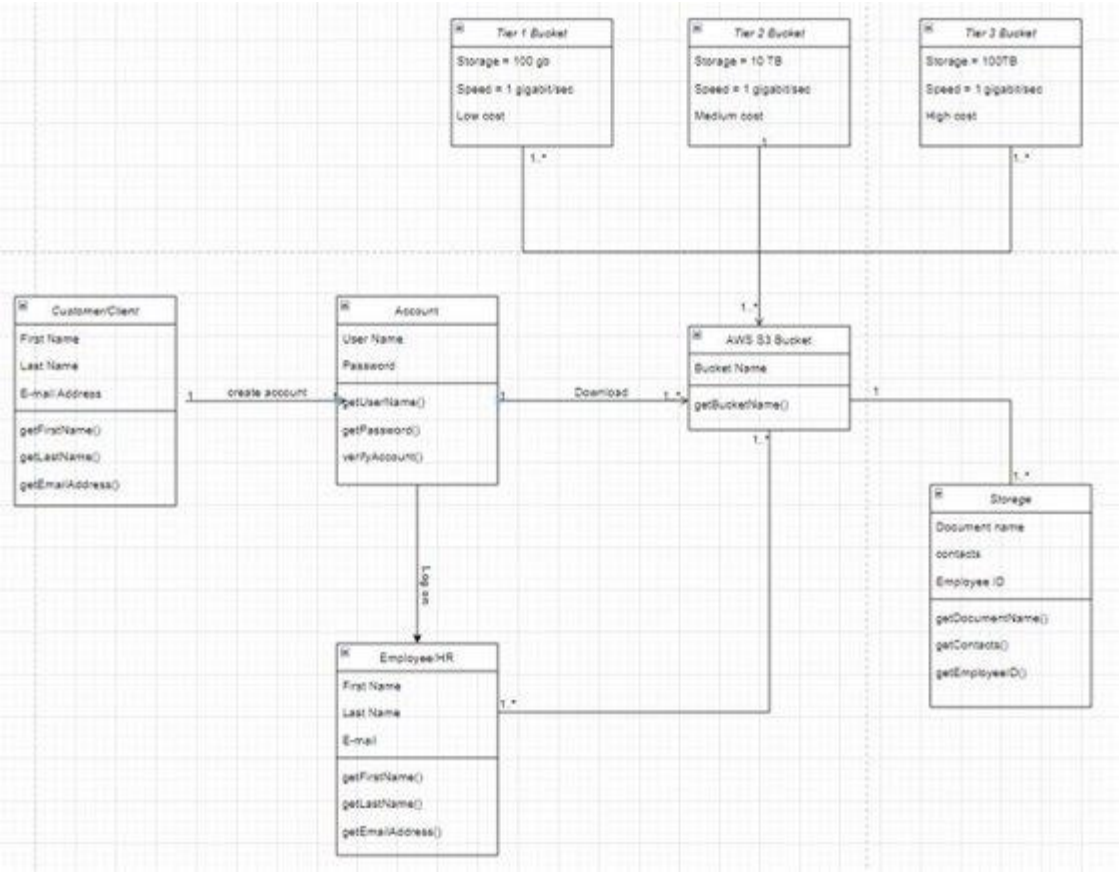
1. Authentication issues for users. By means, there are risks of users getting locked out of their account due to invalid sign in attempts. In order to avoid this risk, it is recommended for users to manually type in email addresses, usernames, and most importantly password; auto fill functionality is not recommended.
2. Data Loss between Memory Manage System (MMS) and AWS Buckets. This can happen when you experience hardware failure. To mitigate this, implement a redundant backup drive, so we have physical data storage in addition to cloud-based storage servers. In addition, use TCP routing protocols between end points on our network to ensure data transmission.
3. Cost can add up. Although this service is a pay per use, meaning you pay only for what you use, the amount can add up. To avoid this, make sure to

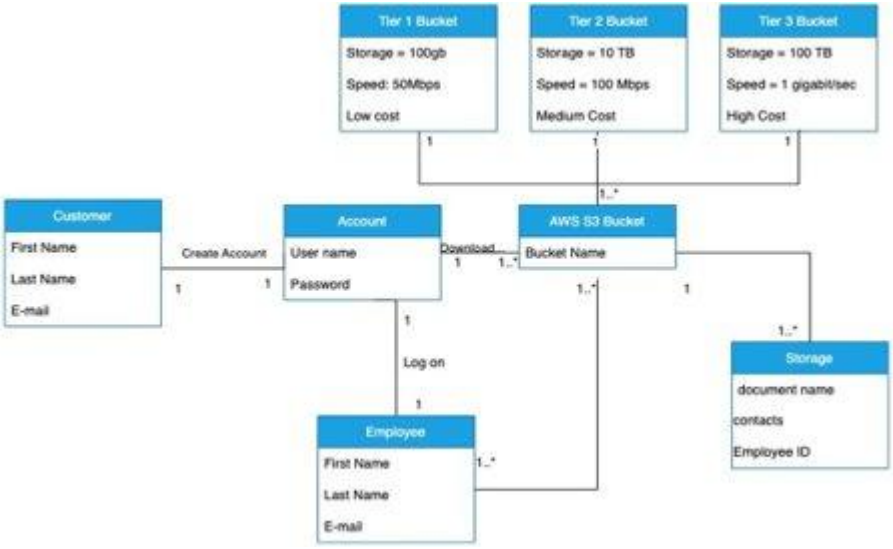
check the bucket size, choosing the right availability zone as different zones have different prices, and it is recommended to store data stored in S3 to EBS.
How to Mitigate: Streamline development process by hiring contractors to create the project and retaining fewer employees for maintenance.

4. Cyber-attack: AWS S3 are preys to ransomware attacks.
   How to Mitigate: Backup and file management are the dynamic duo of ransomware attack solutions. In the event of an attack, these can be used to restore systems and recover data quickly. This is critical since even if the ransom is paid, there is no guarantee that the files will be fully restored.
5. Social Engineering Attack: risk of an employee falling into a phishing attack results in hackers acquiring user accounts and data.
   How to Mitigate: Use best practice and anti-patterns to avoid social engineering attacks.

Updated Domain Model & Class Model

In the Amazon S3 Bucket, users are customers and employees. Customers can make an account to download objects or files into the S3 Bucket. Employees have accounts that allow them to download, update files, and upload files. Customer is the Creator of Account. Account fills the Role of Controller, controlling the behavior of the AWS S3 Bucket and relaying requests. AWS S3 Bucket serves as an Adapter between for Storage. AWS S3 Bucket serves as a Factory for new Tier I, II, and III Buckets. Like most Factory classes, AWS S3 also serves as a Singleton for the MMS.

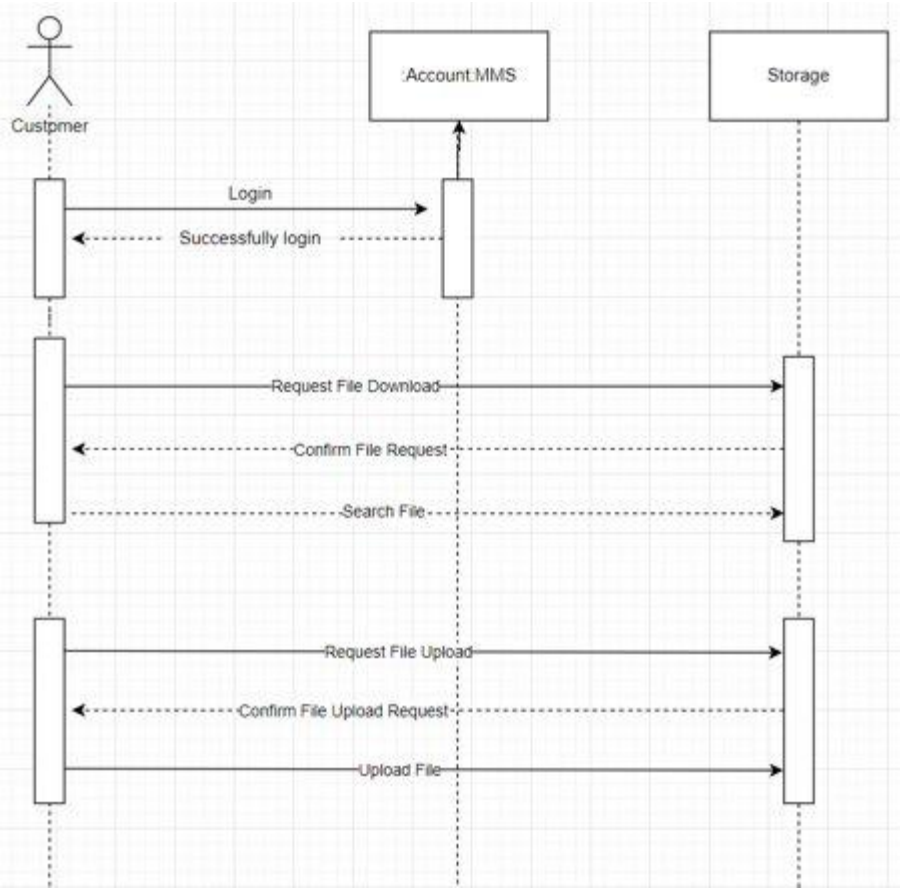




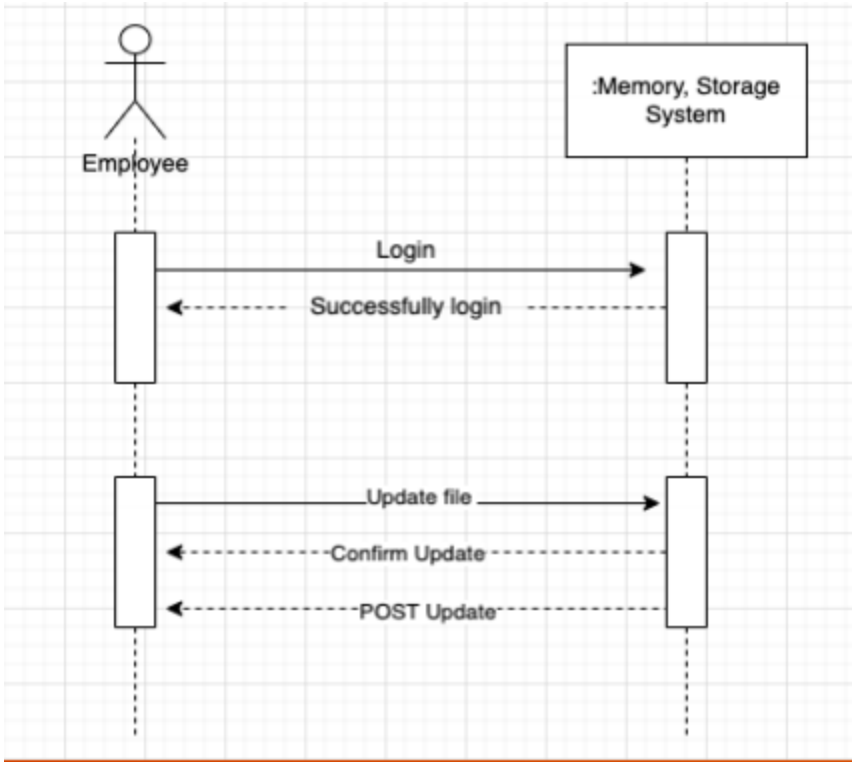
Updated Data Model

Class	Attributes	Data types
Customer/Client	First Name Last Name E-mail	String String Varchar(30)
Tier 1 Bucket	Storage Speed Low cost	Int Double Double
Tier 2 Bucket	Storage Speed Medium cost	Int Double Double
Tier 3 Bucket	Storage Speed High cost	Int Double Double
Account	User name Password	Varchar(30) Varchar(30)
AWS S3 Bucket	Bucket Name	String
Employee/HR	Item 1 Item 2 Item 3	Int Int Int
Storage	Document Name Contacts Employee ID	String Int Varchar(30)

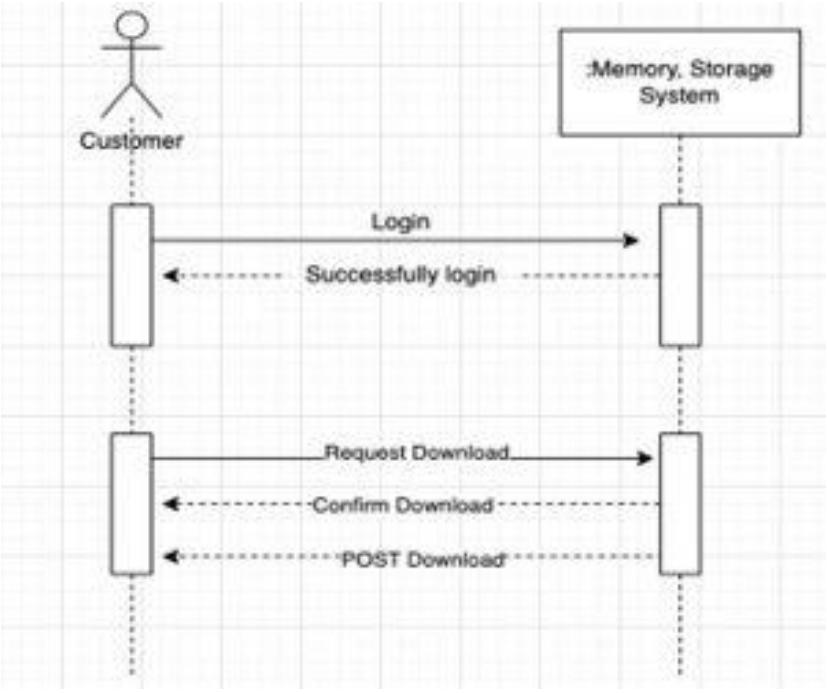
Sequence Diagram for Memory Storage System Model



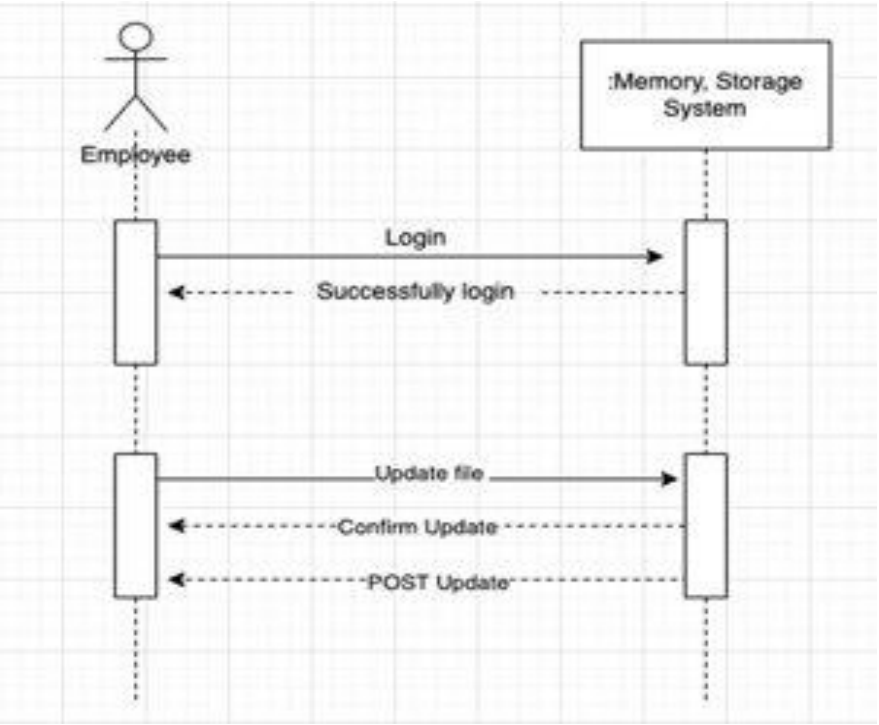
System Sequence Diagram For Customer Upload Files



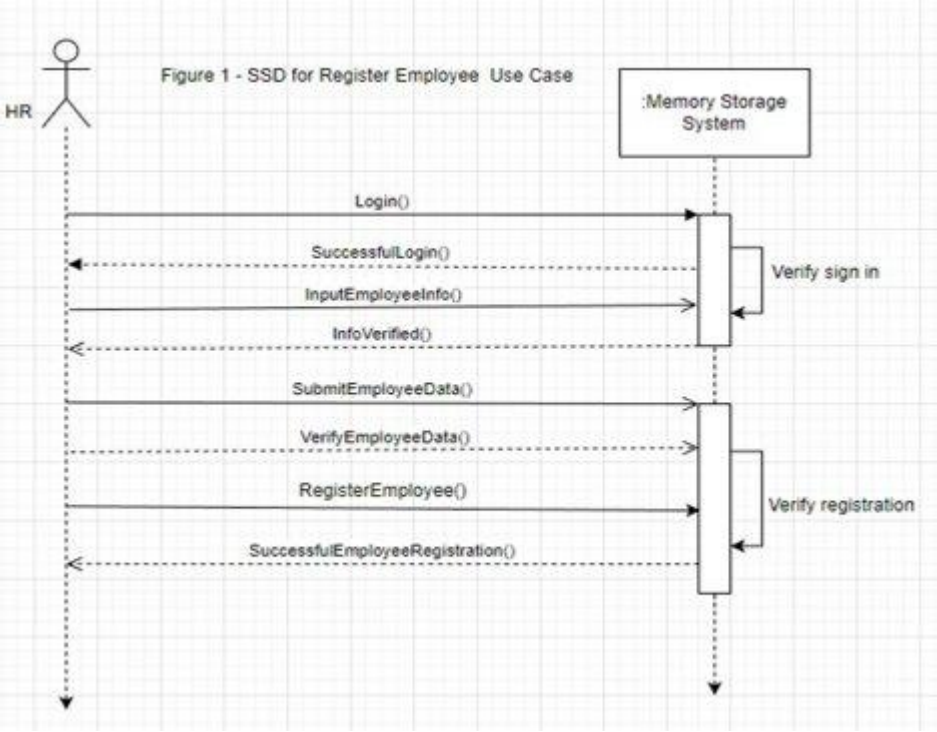
System Sequence Diagram For Customer Download Files



System Sequence Diagram For Employee Upload Files



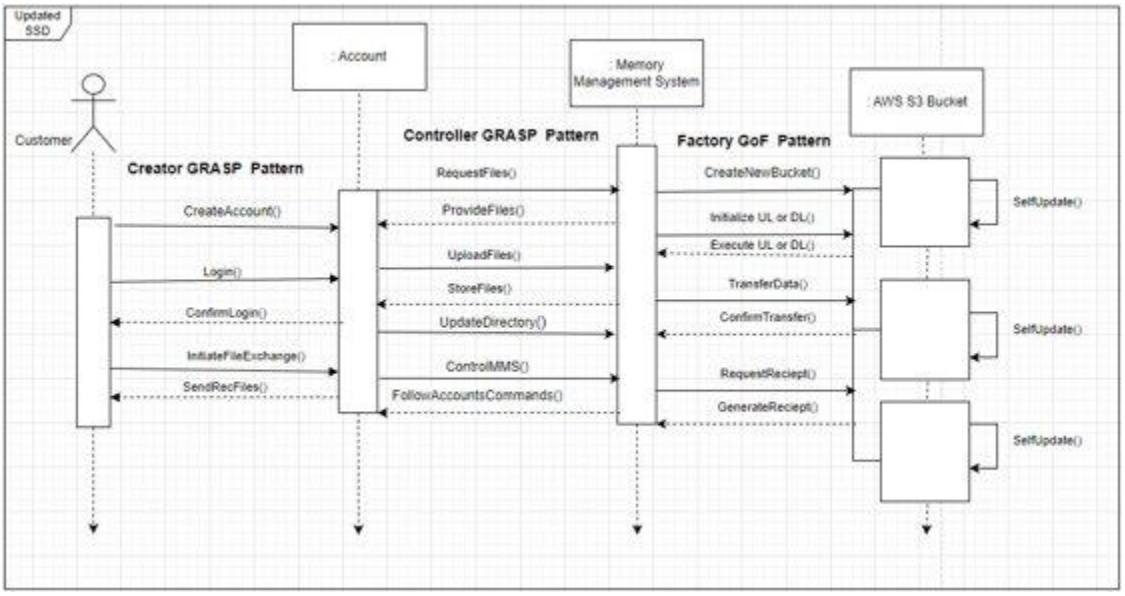
System Sequence Diagram for Register Employee



Creator, Controller & Factory

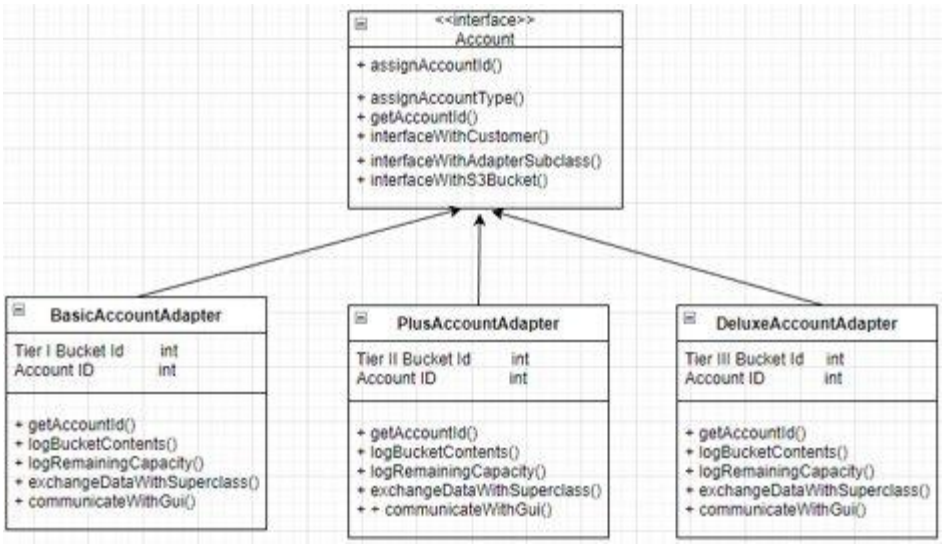
Creator: The Customer class creates instances of Account and its subclasses, Basic, Plus, and Deluxe, filling the Creator role. Customer's singular role in our domain model is to create the Account and specify account details. Once created, the account becomes the Memory Management's major Actor.

Controller: Once an account is initialized by one of our Customers in the Memory Management System, the Account assumes the role of Controller by managing system requests to and from the S3 Bucket. The Account is a non-user object that receives and sends requests for data between the user and S3 Buckets, which store large amounts of data.



Polymorphism & Adapter

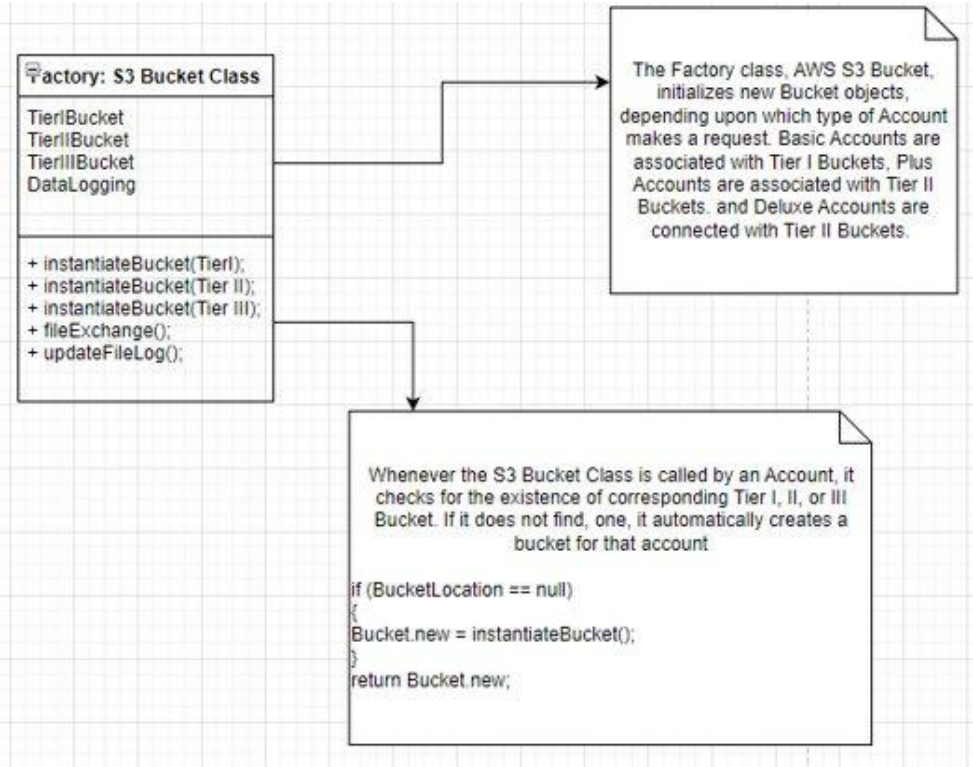
The Polymorphism GRASP pattern applies to our three subclasses Basic, Plus, and Deluxe extending the superclass Account. These subclasses share the basic blueprint for Account, but each varies in memory storage size and cost to the customer. This relationship also represents the Adapter pattern. Whenever a variation of an account is needed, the subclasses serve as Adapters.



Using the Polymorphism GRASP pattern, the Account type adapters for Plus, Basic, and Deluxe inherit Account's blueprint, but have their own variations in storage capacity and data transfer speed.

Factory

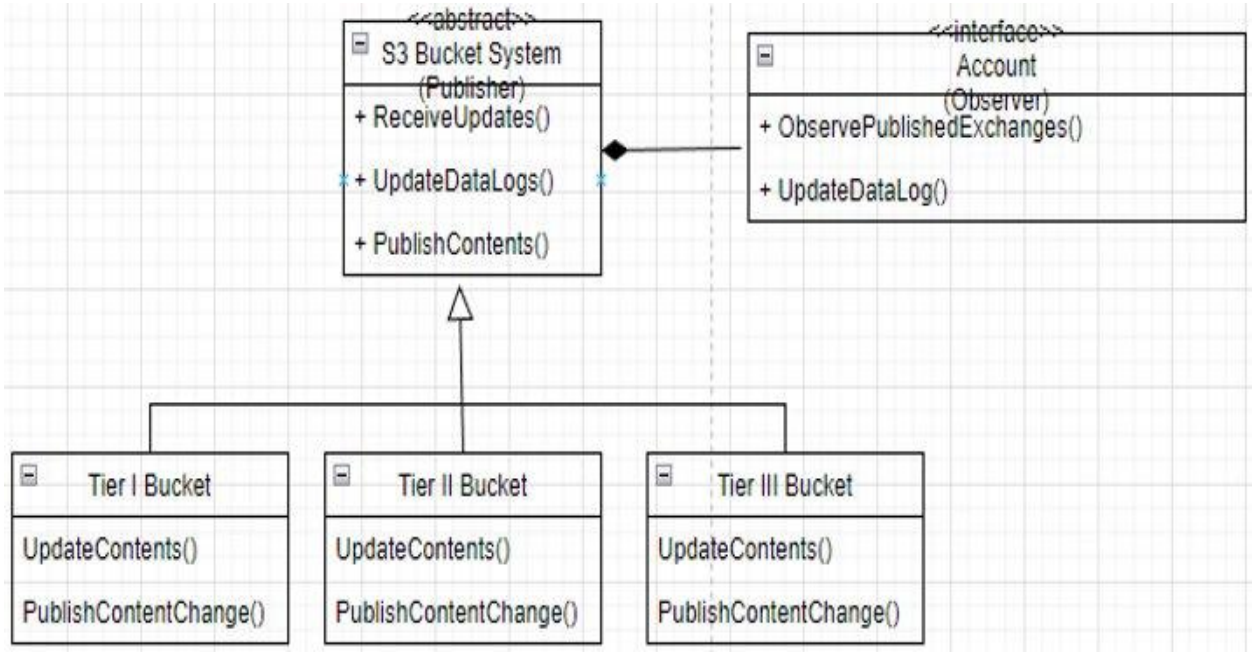
The S3 Bucket class models the GoF pattern known as Factory. That means it creates instances of classes In response to requests it receives from the account class, it manufactures individual bucket objects without exposing creation logic to the client. Depending on requests made by an Account object, the S3 Bucket class can create three distinct object types: Tier I Bucket, Tier II Bucket, Tier III Bucket.



Singleton
Our S3 bucket system represents the GoF pattern for Singleton. The Memory Management System will always employ a single instance of the S3 Bucket System, which communicates with all of our buckets and accounts.

Facade
From the Customer's standpoint, the Account represents the entire Memory Management System, making the Account and our forward-facing GUI a Façade for the entire Memory Management System.

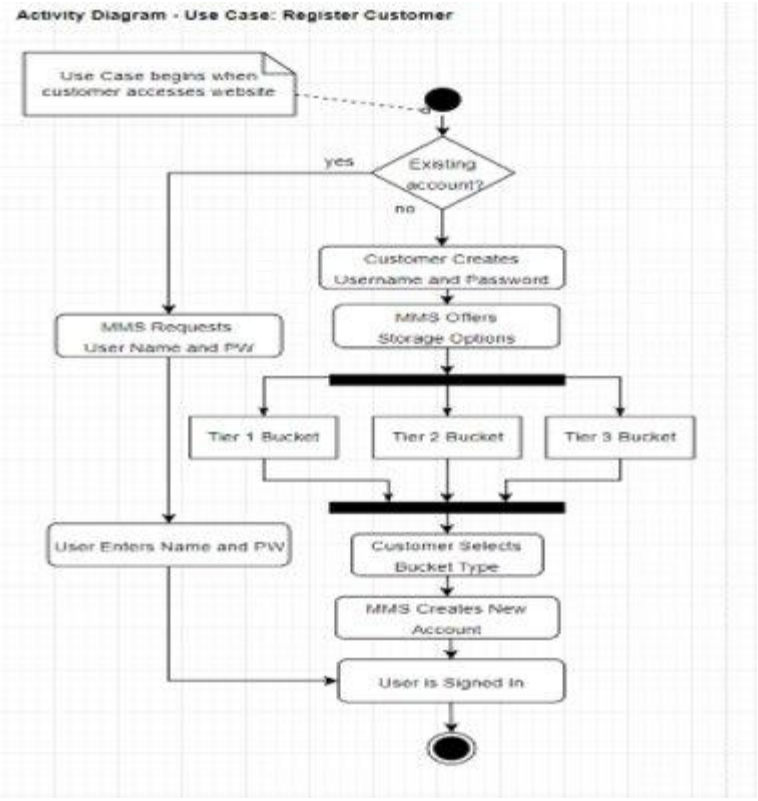
Observer
Each instance of Account keeps a virtual log of the contents of its corresponding Bucket so that it can show users which files are available to retrieve and manipulate. Whenever an Account initiates a change in the contents of its corresponding Bucket, the AWS S3 Bucket publishes an content update to Account. Account maintains a log of the Bucket's contents based on changes it observes that are published by the S3 Bucket System.



Account objects observe updates published by the S3 Bucket System. The S3 Bucket System observes content updates from each of the three bucket types. Account updates its log of contents for the appropriate bucket.

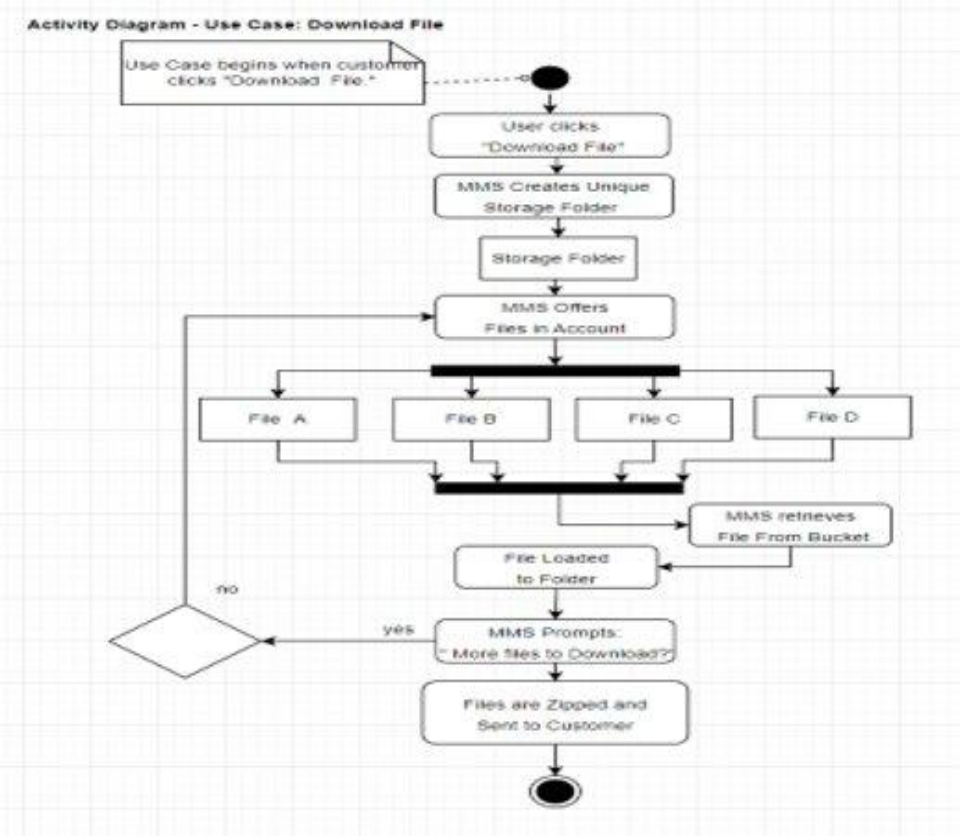
Use Case: Register Customer

In the Activity Diagram above, see how a user creates a new account with the Memory Management System. First MMS checks to see if an account exists if it does it prompts the user for a name and password. The system prompts the user to create a new password and select one of three storage options ranging from tier I to tier III. Once a tier is selected, the MMS automatically creates a corresponding bucket object and account, then links these two with the user's name and password.



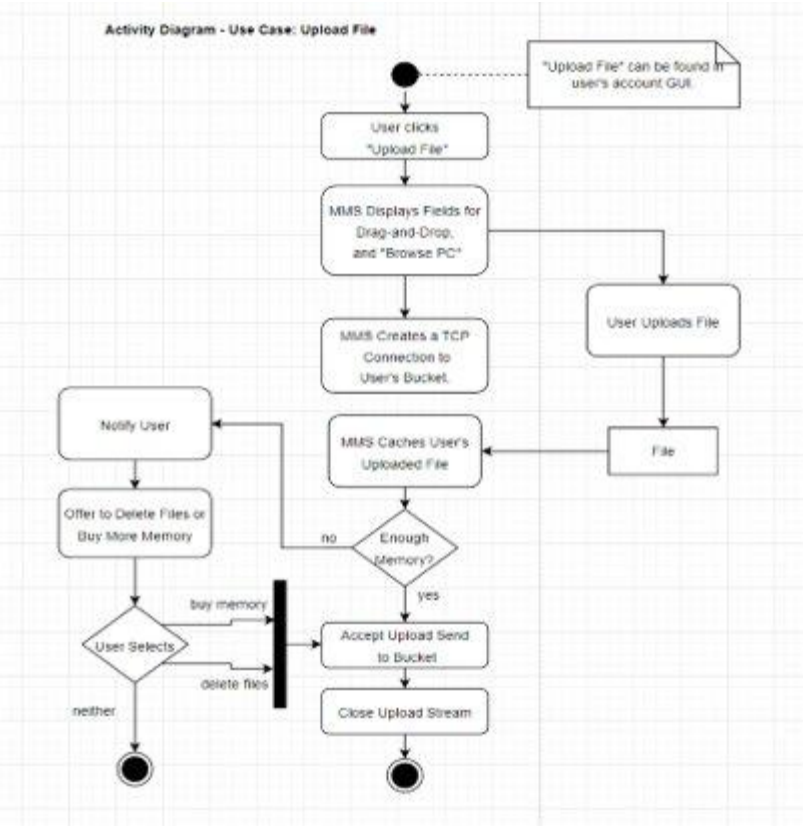
Use Case: Download File

The above activity diagram shows MMS's behavior for the use case "Download File." When the customer initiates a file download, MMS immediately creates a unique storage folder object that will hold the data in zipped format. Next, the MMS offers a list of the user's files. Selected files are loaded into the zip folder. When the user has completed the file selection process, the MMS retrieves the chosen files from the storage bucket, and zips them into a folder that is then sent to the user.



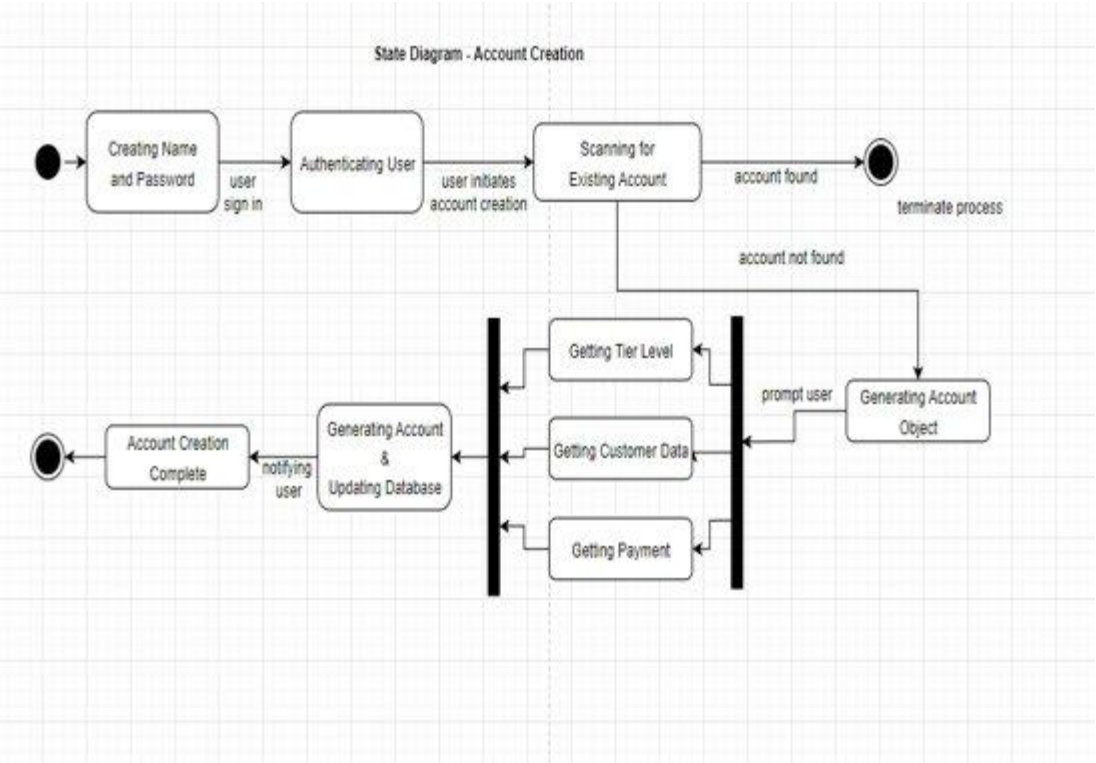
Use Case: Upload File

The above activity diagram shows MMS's behavior for the use case "Upload File." When the customer initiates a file upload, the MMS immediately displays a file upload field with the option to either drag and drop or to browse the user's pc. It then locates and opens a secure TCP connection to the user's unique storage bucket. MMS accepts the user's file selection and evaluates the file size against the user's available storage space. If the user has enough space, the file is transferred to their bucket. If not, the user is prompted to delete files, purchase more space, or cancel the transaction.



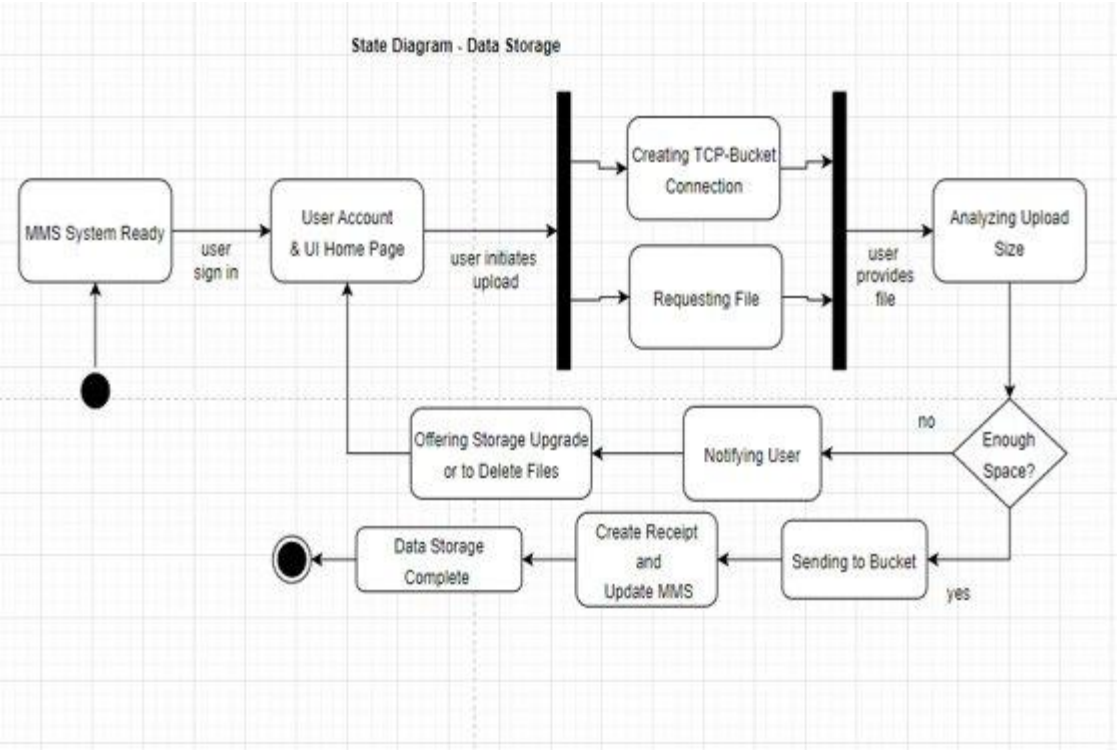
State Diagram: Account Creation

The above state diagram shows MMS's behavior for the account creation process. Once a user creates their name and password, the Memory Management System offers them the chance to create a paid storage account. When the user initiates account creation, the MMS automatically scans for an existing account with the user's identifying information. If it doesn't find one, the MMS presents fields for the user to provide more info. When the user's info is submitted, three smaller menus appear in order, asking the user for their desired tier level, their payment information, and additional contact information and preferences. The final step is to load the Account object with the data provided and update the MMS database accordingly.



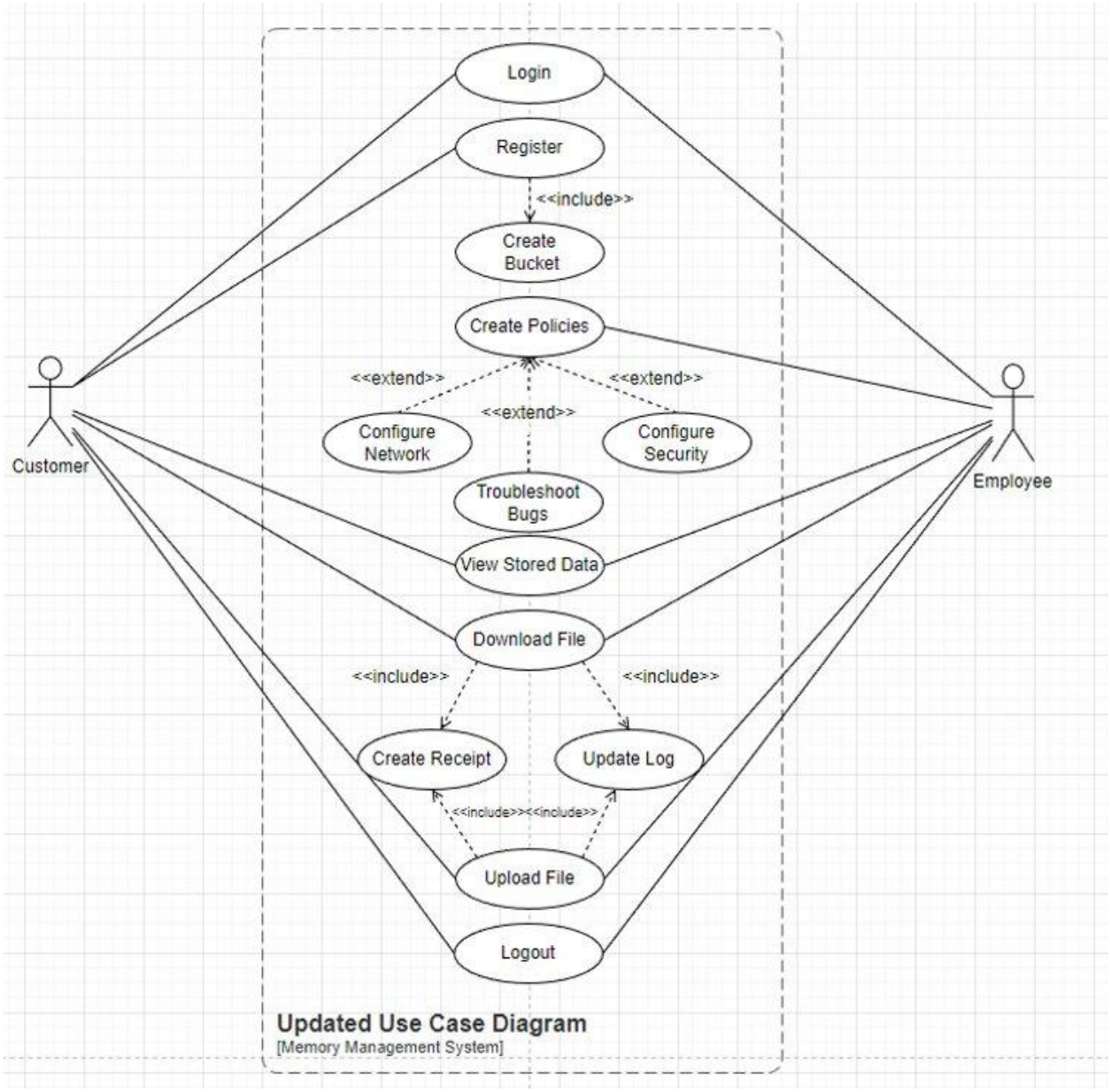
State Diagram: Data Storage

The above state diagram shows MMS's process for storing new data. This process is initiated by the user from his unique Account homepage. Once the user initiates an upload, the MMS simultaneously creates a connection to the user's individual storage container and prompts the user to upload one or more files. Once the MMS receives a file, it decides if the user has purchased enough storage space. If yes, the file is sent to storage, and the transaction is recorded. If no, the user is notified and given the choice of creating more space or canceling the upload request.



Updated Use Case Diagram With Include and Extend

We have updated our original use case diagram to include "include" and "extend" relationships between several of our use cases. The original structure stays the same, giving the user a linear sequence through registering an account creation and uploading and downloading files. Registering an account is now always accompanied by creating a storage bucket object via an include relationship. The Employees' Create Policies use case now has three extension relationships branching out, granting the additional system administration duties of Configuring the Network, Troubleshooting Bugs, and Configuring Security. The final update that we've added to Elaboration 3 is that both use cases: Download File and Upload File include creating transactional receipts an updating the log of files held by the Memory Management System.


Login
Register
<<include>>
Create Bucket
Create Policies
<<extend>>
<<extend>>
<<extend>>
Configure Network
Configure Security
Troubleshoot Bugs
View Stored Data
Download File
<<include>>
<<include>>
Create Receipt
Update Log
<<include>><<include>>
Upload File
Logout
Customer
Employee
Updated Use Case Diagram
[Memory Management System]