Team organization:

Members - subsystems

Max Delnevo (Group leader)	19339653	Web-based application for desktop or tablet for officers of uni-specific student unions to manage and operate the union.
Benjy Hollister	19343388	A mobile app to run on smart phones for students to participate in the student union's activities.
Ivan Kovalchuk	19324937	USU Operation system application platform for web and tablet
Salih Mirza Dogan	19334208	a mobile app to run on mobile phones and on desktop or tablet computers for society leader to organise and operator societies.

Collaboration mechanisms:

URL to project:

https://github.com/IvanKovalchuk-Brookes/COMP5047-Applied-Software-Engineering-Coursework-Group-1

Folders and explanations:

- 19334208 - Salih
    - Activity_diagram
        - Papyrus files for Activity Diagram
    - ClassDiagram
        - Papyrus files for Class Diagram
    - ComponentDiagram
        - Papyrus files for Component Diagram
    - Sequence diagram
        - Papyrus files for Sequence Diagram
    - UseCase
        - Papyrus files for UseCase diagram
    - papyrus
        - Includes
        - (ActivityDiagram,ClassDiagram,SocietyLeaderApp,SocietyLeaderApp1,Societyleaderapp55 all versions)
    - Pictures
        - PNG files for all diagrams

- 19324937 - Ivan
    - ArchitectureDesign
        - Papyrus diagram for Architecture Design
    - PapyrusDiagram
        - Includes:
            - Papyrus files for UseCase
            - Papyrus files for ActivityDiagram
            - Papyrus files for ComponentDiagram
            - Papyrus files for ClassDiagram
    - Pictures
        - PNG files of all completed diagrams
    - WeeklyNotes
        - Collection of all meetings notes
    - QualityRequirements document
- 19339653-Max
    - ActivityDiagram
        - Papyrus files for Activity Diagram
    - ClassDiagram
        - Papyrus files for Class Diagram
    - ComponentDiagram
        - Papyrus files for Component Diagram
    - Pictures
        - PNG files of all completed diagrams
    - SequenceDiagram
        - Papyrus files for Sequence Diagram
    - Usecase
        - Papyrus files for Use case Diagram
    - Quality Requirements document
- 19343388-Benjy
    - Activity Model
        - Papyrus files for Activity Model
    - Class Diagram
        - Papyrus files for Class diagram
    - Component Images
        - Pictures of papyrus diagrams
    - Sequence Diagram
        - Papyrus files for Sequence diagram
    - Software Architecture
        - Papyrus files for Software architecture diagram
    - Use Case
        - Papyrus files for Use Case diagram
    - 5047 CW Quality Requirements (1).docx
- ArchitectureDiagram

    - Papyrus files for Architecture Diagram
- MeetingNotes
    - Collection of all meeting notes across the project

Team operation:

Date	Venue	Start	Finish	Topics	Chairperson	Attendance	Agenda url	Minutes url
23/09/2025	AB 1.15	12:45	1:45	1.Whatsapp Group created and 3 members added 2.Main Group github repository created 3. Personal github repositories created 4. Subsystems selected by members, (p2=1,p1=2,p3=3)(Default p4=4)	Max Delnevo	Attendees: 19339653, Max Delnevo (p1) 19343388, Benjy Hollister (p2) 19324937, Ivan Kovalchuk (p3) Absentees: 19334208, Salih Mirza Dogan (p4)	N/A	Link
30/09/2025	AB 1.15	12:50	1:45	1.Get Papyrus working and figure out linking with Git 2.Discuss quality requirements regarding the sections	Benjy Hollister	19339653, Max Delnevo 19343388, Benjy Hollister 19324937, Ivan Kovalchuk	Link	Link
07/10/2025	AB 1.15	12:45	1:00	1.Discussed Quality requirements work - Faculty 2.Looked at	Salih Mirza Dogan	19339653, Max Delnevo 19334108, Salih Mirza Dogan	Link	Link

				creating UML diagrams on Papyrus				
14/10/2025	AB 1.15	12:10	1:00	1.Discuss current location in coursework 2.Defined plan for next week - Finalise quality requirements - Finalise Use Case model - Finalise Activity model 3.Take a look at activity model work	Ivan Kovalchuk	19339653, Max Delnevo 19343388, Benjy Hollister 19324937, Ivan Kovalchuk	Link	Link
22/10/2025	JHB Main Hall	11:10	12:10	1. Discussed Quality requirements 2. Discussed UML diagrams 3. Discussed Activity diagrams 4. Uploaded to GitHub individual branches	Max Delnevo	19339653, Max Delnevo 19343388, Benjy Hollister 19324937, Ivan Kovalchuk	Link	Link
28/10/2025	AB 1.15	12:10	12:40	1.Discuss current progress on the work (Quality requirements and UML diagrams) 2.Discuss the Architecture	Benjy Hollister	19339653, Max Delnevo 19343388, Benjy Hollister	Link	Link

				System lecture. Currently think about how the system would link together as no enough info to make the diagrams provided yet.				
04/11/2025	AB 1.15	12:10	13:00	1.p1 noticed that work allocation has been incorrect. Will redo for the correct allocation (worked on part 4 not part 2). 2.Discussed work for next week. Architectural design of subsystems. Focusing on the microservices of the subsystem and the ones it requests from the other subsystems. 3.Think about how the Subsystems come together. Attempt to design the architectural design of the whole system once we have our microservice diagrams completed	Ivan Kovalchuk	19339653, Max Delnevo 19343388, Benjy Hollister 19324937, Ivan Kovalchuk	Link	Link

11/11/2025	AB 1.15	12:12	13:00	1.Discussed the tidy up of the GitHub repository 2.Looked at overall submission of coursework, Only a few UMLs left until complete (and team reports) 3.Discussed architecture review. Will aim to have all members present to put together the architecture UML 4.Deadline for architecture design is next week (18/11/2025)	Max delnevo	19339653, Max Delnevo 19324937, Ivan Kovalchuk	Link	Link
25/11/2025	AB 1.15	12:30	13:15	1.Discuss what work will be needed for final submission 2.Reviews over Component diagram theory 3.Reviews over Component diagram documentation 4.Plan to have a follow up meeting for architecture design to be put together	Benjy Hollister	19339653, Max Delnevo 19343388, Benjy Hollister 19324937, Ivan Kovalchuk	Link	Link

02/12/2025	Abercrombie	12:15	17:35	1.Group members collected component diagrams together to make an Architecture review. Last diagram required by Ivan, Files for the diagram and PNG are uploaded to the GitHub shared repo. 2.Discussed component diagrams. 3.Discussed team report. 4.Set deadline for all PDFs to be completed and on the GitHub group repo by this Thursday (4/12/2025) 5.Team report with architecture review to be completed by the same deadline. 6.Set deadline for file structures to be completed on team report.	Max Delnevo	19339653, Max Delnevo 19343388, Benjy Hollister 19324937, Ivan Kovalchuk 19334108, Salih Mirza Dogan	Link	Link

Statement on collaboration:

Discussing which documents were checked by whom and at which meetings

Over the course of the project, it was mainly theoretical discussions over the work that took place in the meetings. However there were a few instances that work was presented and discussed for example; during week 4 quality requirements and use case diagrams were shown for 3 of the members and improvements were discussed, this was followed up during week 5 where they were shown again with the members going away and tweaking their designs.

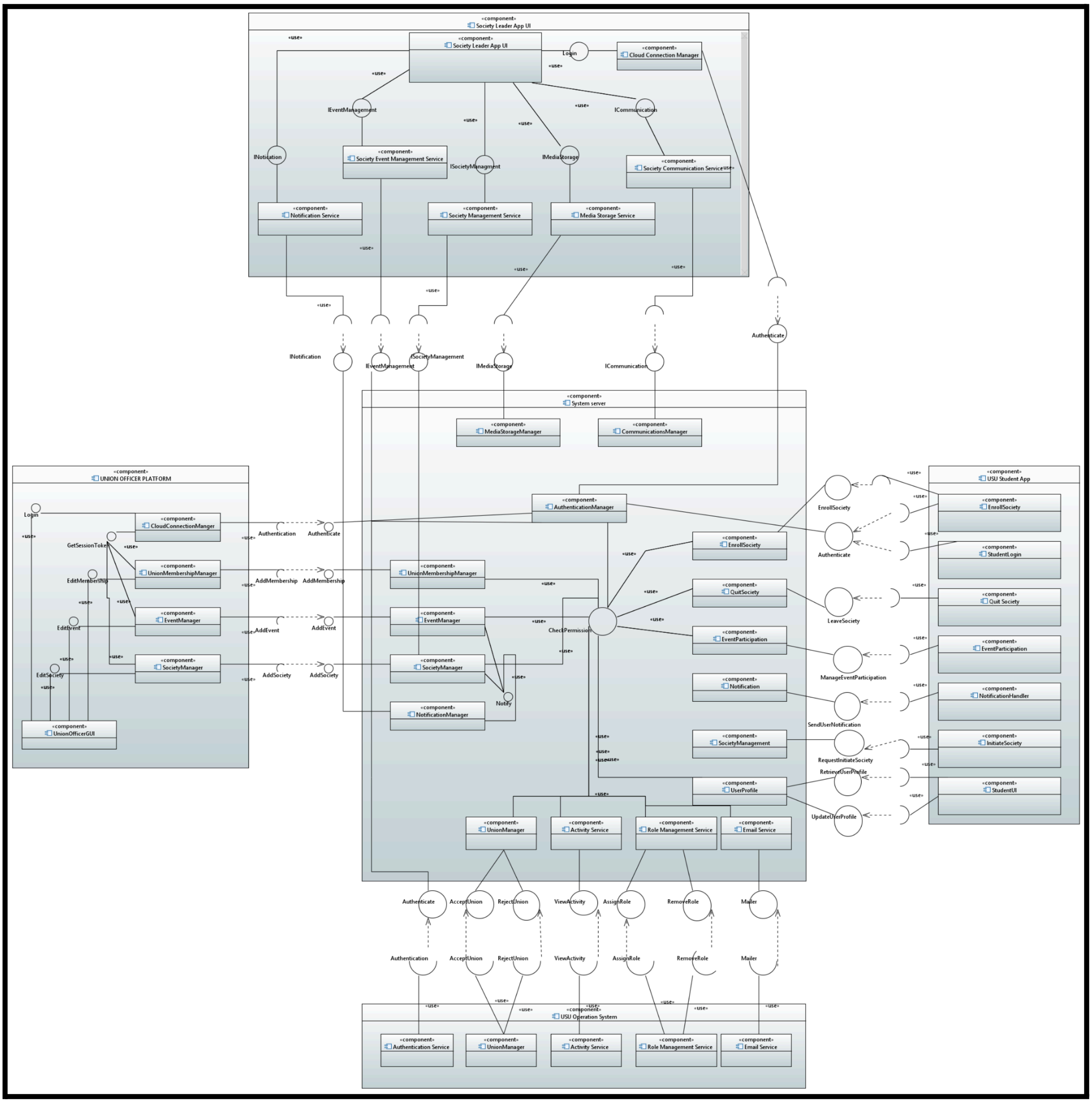
During week 6, Activity diagrams had been shown and discussed while then going on to discuss how the system would be integrated and give thought to how the architecture of the system would be integrated.

During week 7 it became apparent that a member of the group had done work over the wrong subsystem that was not allocated to them. That user took it upon themselves to then correct this by redoing all the work up to this point again. All the previous incorrect work had been removed.

In the final week (meeting 10). All members have had their work reviewed as each member starts to add their component diagrams together for the architecture review. GitHub has had its repository tidied and a deadline has been set for Thursday for everything to be placed into a PDF by the deadline ready to be submitted.

Architectural design:

Diagram


«component» Society Leader App UI
«component» Society Leader App UI
«component» Cloud Connection Manager
Login
IEventManagement
ICommunication
INotication
ISocietyManagment
IMediaStorage
«component» Society Event Management Service
«component» Society Communication Service
«component» Notification Service
«component» Society Management Service
«component» Media Storage Service
«use»
INotification
IEventManagement
ISocietyManagement
IMediaStorage
ICommunication
Authenticate
«component» System server
«component» MediaStorageManager
«component» CommunicationsManager
«component» AuthenticationManager
«component» UNION OFFICER PLATFORM
Login
GetSessionToken
EditMembership
EditEvent
EditSociety
«component» CloudConnectionManger
«component» UnionMembershipManager
«component» EventManager
«component» SocietyManager
«component» UnionOfficerGUI
Authentication
Authenticate
AddMembership
AddMembership
AddEvent
AddEvent
AddSociety
AddSociety
«component» UnionMembershipManager
«component» EventManager
«component» SocietyManager
«component» NotificationManager
Notify
CheckPermission
«component» EnrollSociety
«component» QuitSociety
«component» EventParticipation
«component» Notification
«component» SocietyManagement
«component» UserProfile
«component» UnionManager
«component» Activity Service
«component» Role Management Service
«component» Email Service
EnrollSociety
Authenticate
LeaveSociety
ManageEventParticipation
SendUserNotification
RequestInitiateSociety
RetrieveUserProfile
UpdateUserProfile
«component» USU Student App
«component» EnrollSociety
«component» StudentLogin
«component» Quit Society
«component» EventParticipation
«component» NotificationHandler
«component» InitiateSociety
«component» StudentUI
Authenticate
AcceptUnion
RejectUnion
ViewActivity
AssignRole
RemoveRole
Mailer
Authentication
AcceptUnion
RejectUnion
ViewActivity
AssignRole
RemoveRole
Mailer
«component» USU Operation System
«component» Authentication Service
«component» UnionManager
«component» Activity Service
«component» Role Management Service
«component» Email Service

Documentation:

UnionOfficerPlatform:

Component	UnionOfficerGUI

Description	Runs on application to provide the user interface for union officers to access the platform.
Stereo Type	Component
Required Interfaces	N/A
Provided Interfaces	-Login -EditEvent -EditUnionManagment -EditSociety

Component	CloudConnectionManager
Description	Runs on application to authenticate components and user logging in.
Stereo Type	Component
Required Interfaces	-Login
Provided Interfaces	-GetSessionToken -Authentication

Component	EventManager
Description	Runs on application to send requests to the cloud to manage the event details.
Stereo Type	Component
Required Interfaces	-GetSessionToken
Provided Interfaces	-EditEvent -AddEvent

Component	UnionMembershipManager
Description	Runs on Application to request cloud to make changes in status and data in union membership.
Stereo Type	Component
Required Interfaces	-GetSessionToken
Provided Interfaces	-EditUnionManagment -AddMembership

Component	SocietyManager
Description	Runs on application to send requests to the cloud to manage the society details.
Stereo Type	Component
Required Interfaces	-GetSessionToken
Provided Interfaces	-EditSociety -AddSociety

SystemServer:

Component	AuthenticationManager
Description	Runs on cloud to authenticate users and services.
Stereo Type	Component - Service
Required Interfaces	-Authentication
Provided Interfaces	-CheckPermission

Component	Event manager
Description	Runs on the cloud to process the changes to events.
Stereo Type	Component - Service
Required Interfaces	-CheckPermission
Provided Interfaces	-Add Event

Component	Union membership manager
Description	Runs on cloud to process the changes to union membership.
Stereo Type	Component - Service
Required Interfaces	-CheckPermission
Provided Interfaces	-Add Membership

Component	Society manager

Description	Runs on cloud to process the changes to society.
Stereo Type	Component - Service
Required Interfaces	-CheckPermission
Provided Interfaces	-Add Society

Component	Notification manager
Description	Runs on Cloud to send notifications of changes.
Stereo Type	Component - Service
Required Interfaces	N/A
Provided Interfaces	-Notify

Interfaces:

Name	Login	
Provider	CloudConnectionManager	
Operation	Signature	Login(UnionOfficerID)
	Function	Provides ID of union officer and logs into their account.

Name	Authenticate	
Provider	AuthenticationManager	
Operation	Signature	Authenticate(UnionOfficerID)
	Function	Passes login information across to the server to confirm the details are correct.
Operation	Signature	Authenticate(StudentID)
	Function	Logs the student into the USU system and return an access token

Name	GetSessionToken	
Provider	CloudConnectionManager	
Operation	Signature	CreateSessionToken()
	Function	Creates a session token for each other component on the officer platform.

Name	EditEvent	
Provider	EventManager	
Operation	Signature	EditEvent(EventID)
	Function	Updates the event details at the ID provided

Name	AddEvent	
Provider	EventManager	
Operation	Signature	AddEvent()
	Function	Sends a request to EventManager to create a new event.

Name	EditUnionMembership	
Provider	UnionMembershipManagement	
Operation	Signature	EditUnionMembership(UnionID)
	Function	Updates the membership details of a union at the provided ID

Name	AddMembership	
Provider	UnionMembershipManager	
Operation	Signature	AddMembership()

	Function	Sends a request to the server to add a membership entry to the database.

Name	EditSociety	
Provider	SocietyManager	
Operation	Signature	EditSociety(SocietyID)
	Function	Sends the update request to the society manager where the ID is provided.

Name	AddSociety	
Provider	SocietyManager	
Operation	Signature	AddSociety()
	Function	Sends a request to the server to create a new society.

Name	CheckPermission	
Provider	AuthenticationManager	
Operation	Signature	CheckPermission()
	Function	Confirms permissions are accepted before allowing service components to access data.

Name	Notify	
Provider	SocietyManager EventManager	
Operation	Signature	NotifySociety()
	Function	Sends out notifications regarding updates to the society to requirement members.

Operation	Signature	NotifyEvent()
	Function	Sends out notifications regarding events to a maillist.
Operation	**Signature**	sendEventUpdate()
	Function	Sends update to students who are subscribed to an event

Society Leader App UI

Component	Society Leader App
Description	The Society Leader App UI frontend interface sued by society leaders to manage events, update society information, upload media, and communicate with members. It interacts with all microservices through their interfaces
Stereo Type	Component
Required Interfaces	IEventManagement ISocietyManagement IMediaStorage ICommunication INotification Authenticate
Provided Interfaces	-None

Component	Notification Service
Description	for managing and delivering notifications to users within the USU system. Whenever a Society Leader creates or updates an event, this service is triggered to inform all subscribed students.
Stereo Type	Component -service
Required Interfaces	-N/A
Provided Interfaces	-INotification

Component	Society Event Management Service
Description	responsible for managing the full lifecycle of society events.
Stereo Type	Component -service
Required Interfaces	-IEventManagement,SocietyManagement
Provided Interfaces	-IEventManagement

Component	Society Management Service
Description	Backend microservice responsible for handling all operations related to managing a society's core information
Stereo Type	Component -service
Required Interfaces	-ISocietyManagement
Provided Interfaces	-ISocietyManagement

Component	Media Storage Service
Description	ackend microservice responsible for storing and retrieving media files such as images and videos uploaded by society leaders. It provides a unified interface for managing media assets and ensures secure, efficient access for other components.
Stereo Type	Component -service
Required Interfaces	-N/A
Provided Interfaces	-IMediaStorage

Component	Society Communication Service

Description	Responsible for enabling communication between society leaders and their members
Stereo Type	Component -service
Required Interfaces	-N/A
Provided Interfaces	-ICommunication

Component	Cloud Connection Manager Service
Description	Provides authentication for the subsystem by verifying user identity and access permissions through the Authenticate interface. It acts as the secure entry point to the environment used by all USU services
Stereo Type	Component -service
Required Interfaces	-N/A
Provided Interfaces	-Authenticate

Component	MediaStorageManager
Description	Component handles the storage and retrieval of media files within the system server. It processes media related requested coming from other components
Stereo Type	Component -service
Required Interfaces	-IMediaStorage
Provided Interfaces	-None

Component	CommunicationManager
Description	Manages message delivery, ıt processes communication requests from other components and ensures that messages are routed correctly
Stereo Type	Component -service

Required Interfaces	-ICommunication
Provided Interfaces	-None

Component	EventManager
Description	Runs on the cloud to process the changes to events.
Stereo Type	Component -service
Required Interfaces	-IEventManagement
Provided Interfaces	-Notify

Component	SocietyManager
Description	The SocietyManagement component manages society-related operations within the system server.
Stereo Type	Component -service
Required Interfaces	-ISocietyManagement
Provided Interfaces	-Notify

Component	NotificationManager
Description	It processess notification requests from other backend components and ensures that updates, alerts, or event related messages are deliveredn to the appropriate user
Stereo Type	Component -service
Required Interfaces	-INotification
Provided Interfaces	-Notify

Interfaces

Name	INotificationService	
Provider	NotificationManager	
Operation	Signature	sendNotification()
	Function	Sends a notification request to the NotificationManager, which delivers the message to the appropriate users so they are informed about event updates or important society announcements

Name	IEventManagement	
Provider	SocietyManager	
Operation	Signature	CreateEvent(eventData : EventData)
	Function	Sends the event details to the backend to create a new event and returns the created event object.

Name	ISocietyManagement	
Provider	EventManager	
Operation	Signature	GetSocietyInfo(societyID : String)
	Function	Retrieves all stored information about the specified society so it can be displayed or processed by the Society Leader Appt.

Name	IMediaStorage	
Provider	MediaStorageManager	
Operation	Signature	uploadMedia()
	Function	Uploads a media file to the system’s MediaStorageManager,

		where the file is securely stored and made accessible for use in event pages

Name	ICommunication	
Provider	CommunicationManager	
Operation	Signature	sendMessage()
	Function	Sends a message to the CommunicationManager, which routes it to the correct recipient so that society leaders and members can communicate efficiently within the system.

Name	Authenticate	
Provider	AuthenticationManager	
Operation	Signature	authenticateUser()
	Function	**Verifies the user's login credentials through the AuthenticationManager to ensure that only authorised society leaders can access the subsystem.**

Student App Layer

Component	StudentUI
Description	A microservice running on the student app for the students to interact with the system.
Stereo Type	Component
Required Interfaces	UpdateUserProfile, RetrieveUserProfile
Provided Interfaces	

Component	EnrollSociety
Description	A microservice running on the student app to send a request for a user to join an society
Stereo Type	Component
Required Interfaces	Authenticate, EnrollForSociety
Provided Interfaces	

Component	StudentLogin
Description	Handles login workflow by collecting login input from the student and sending authentication requests to the backend service
Stereo Type	Component
Required Interfaces	Authenticate
Provided Interfaces	

Component	QuitSociety
Description	Handles the process of a student leaving/quitting a society, and sends a request to the backend service for the user to be removed from that society
Stereo Type	Component
Required Interfaces	LeaveSociety
Provided Interfaces	

Component	InitiateSociety
Description	A microservice running on the student app to request that a student be permitted to create a society through an application, and then to initiate it on the cloud system if it has been accepted
Stereo Type	Component
Required Interfaces	RequestInitiateSociety
Provided Interfaces	

Component	EventParticipation
Description	A microservice running on the student app to allow students to search for, view, and receive notifications for events
Stereo Type	Component
Required Interfaces	ManageEventParticipation
Provided Interfaces	

Component	NotificationHandler
Description	A microservice running on the student app to manage the delivery and display of notifications for each user based on the societies and events they're involved in.
Stereo Type	Component
Required Interfaces	SendUserNotification
Provided Interfaces	

Cloud System - Services Layer

Component	EnrollSociety
Description	Microservice on the backend that receives the request made by EnrollSociety on the app and starts the enrollment process, including validating the rules, records and checking authentication, before enrolling them onto the society.
Stereo Type	Service
Required Interfaces	CheckPermission, Authenticate
Provided Interfaces	EnrollForSociety

Component	QuitSociety
Description	Microservice on the backend that receives the request made by QuitSociety on the app and starts the leaving a society process
Stereo Type	Service
Required Interfaces	CheckPermission, Authenticate
Provided Interfaces	LeaveSociety

Component	AuthenticationManager
Description	This microservice handles the verification of student credentials, the token generated, and checking of the authentication policies.
Stereo Type	Service
Required Interfaces	
Provided Interfaces	CheckPermission, Authenticate

Component	SocietyManagement Service
Description	Manages the backend operations of all societies, creating, deleting, updating, managing members. Will interact with initiate society on the student app
Stereo Type	Service
Required Interfaces	CheckPermission
Provided Interfaces	RequestInitiateSociety

Component	UserProfile Service
Description	Manages backend operations for student profiles, creating, deleting, updating, event participation.
Stereo Type	Service
Required Interfaces	CheckPermission
Provided Interfaces	UpdateUserProfile, RetrieveUserProfile

Component	Notification Service
Description	Handles creation, management, and delivery of notifications to students, including event reminders, activity updates, system alerts.
Stereo Type	Service
Required Interfaces	

Provided Interfaces	SendUserNotification

Component	EventParticipation Service
Description	Handles requests from the app for users to search, view and receive notifications for events.
Stereo Type	Service
Required Interfaces	
Provided Interfaces	ManageEventParticipation

Specification of Interfaces

Name	EnrollForActivity	
Provider	EnrollActivity	
Operation	**Signature**	joinSociety()
	Function	Is to allow a student to enroll into a society, given they have the permissions of them
Operation	**Signature**	searchSociety()
	Function	Allows a student to search the database for societies that are offered at their university

Name	RequestInitiateSociety	
Provider	SocietyManagement Service	
Operation	**Signature**	submitApplication()
	Function	Lets the student submit an application to create a new society
Operation	**Signature**	trackApplication()
	Function	Student is able to check the approval status of their application by the union.

Name	LeaveSociety

Provider	QuitSociety	
Operation	**Signature**	quitSociety()
	Function	Removes the student from a given society
Operation	**Signature**	
	Function	

Name	UpdateUserProfile	
Provider	UserProfile Service	
Operation	**Signature**	updateUserProfile()
	Function	Update information about the student, for example emails, phone number, etc
Operation	**Signature**	changePreferences()
	Function	Adjusts the student preferences on their profile

Name	RetrieveUserProfile	
Provider	UserProfile Service	
Operation	**Signature**	getUserProfile()
	Function	Returns the stored data for the specified student
Operation	**Signature**	getStudentMemberships()
	Function	List the societies that the student is a part of

Name	ManageEventParticipation	
Provider	EventParticipation	
Operation	**Signature**	searchEvents()
	Function	Allows student to browse or search for events
Operation	**Signature**	registerEvent()
	Function	Registers a student for an event the society is holding

Component	OfficerUI
Description	Web interface used by USU officers to view dashboards, manage unions.
Stereo Type	Component
Required Interface	RetrieveAdminData, UpdateAdminData
Provided Interface	

Component	AuthenticationService
Description	Central Identity provider
Stereo Type	Component
Required Interface	
Provided Interface	

Component	Union Management Service
Description	Core microservice to manage unions
Stereo Type	Component
Required Interface	ModifyUnion, RetrieveUnion
Provided Interface	

Component	Role Management Service
Description	Microservices to manage roles inside each of the union
Stereo Type	Component
Required Interface	UnionMemberList, AssignRole
Provided Interface	

Component	Activity Service
Description	Microservice to view and control union activity on the platform
Stereo Type	Component
Required Interface	UnionList
Provided Interface	

Component	Email Service
Description	Microservice to send email with information
Stereo Type	Component
Required Interface	SendEmail
Provided Interface	

Cloud System:

Component	AuthenticationManager
Description	Validates officer credentials, sessions, and permissions
Stereo Type	Component
Required Interface	Authenticate, CheckPermission
Provided Interface	

Component	UnionManagement Service
Description	Service for managing unions
Stereo Type	Component
Required Interface	ModifyUnion, RetrieveUnion
Provided Interface	CheckPermission

Component	Activity Service
Description	Service for control union activity
Stereo Type	Component
Required Interface	UnionActivity
Provided Interface	CheckPermission