Module Interface Specification for Software Engineering

Team #2, Campus Connections
Waseef Nayeem
Zihao Du
Matthew Miller
Firas Elayan
Abhiram Neelamraju
Michael Kim

January 17, 2024

1 Revision History

Date	Version	Notes
Jan 15	1.0	Add introduction and module decomposition
Jan 17	1.0	Revision 0

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at SRS

2.1 Abbreviations and Acronyms

symbol	description
MIS	Module Interface Specification
MG	Module Guide
SRS	Software Requirement Specification
AR	Augmented Reality

Contents

3 Introduction

The following document details the Module Interface Specifications for CampusConnections. CampusConnections is a social media application with impressive AR camera and real time location map features that allows McMaster University students and visitors have an immersive user experience and expand their social networking. This application allows users to make new friends online and also encourage users to strengthen the friendship by in-person meet-ups with a on-campus location-sharing feature. It also provides heat maps of events and users, which allows students to join the most popular activities on campus. Besides, the application maintainers will share up-to-date events and lectures information for the community. The MIS will detail specifications for the project described above.

Complementary documents include the System Requirement Specifications (SRS) and Module Guide. (MG) The full documentation and implementation can be found at https://github.com/beatlepie/4G06CapstoneProjectTeam2/blob/main/docs/SRS-Volere/SRS.pdf and https://github.com/beatlepie/4G06CapstoneProjectTeam2/blob/main/docs/Design/SoftArchitecture/MG.pdf

4 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol := is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1|c_2 \Rightarrow r_2|...|c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by Software Engineering.

Data Type	**Notation**	**Description**
character	char	a single symbol or digit
integer	$\mathbb{Z}$	a number without a fractional component in (-∞, ∞)
natural number	$\mathbb{N}$	a number without a fractional component in [1, ∞)
real	$\mathbb{R}$	any number in (-∞, ∞)
boolean	$\mathbb{B}$	True or False
sequence of T	<T>	a list of object with type T
asynchronous step T	Task<T>	an asynchronous result of T

The specification of Software Engineering uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In

addition, Software Engineering uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**
Hardware-Hiding	
Behaviour-Hiding	AR Interface Module
	Map Interface Module
	User Module
	Lecture Module
	Event Module
	Account Module
	Permission Module
	User Profile Module
	User Login Module
	Friend Manager Module
	Friend Request Module
	Friend Chat Module
	Lecture Detail View Module
	Event Detail View Module
	Lecture List Manager Module
	Event List Manager Module
Software Decision	Database Module
	Server Module
	Authentication Module
	AR Camera Module
	Mapbox Module
	Activity Detail View Module
	Pagination and Filter Module

Table 1: Module Hierarchy

6 MIS of Account Module

6.1 Module

Account

6.2 Uses

Database Module, User Module, Authentication Module

6.3 Syntax

6.3.1 Exported Constants

None

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
UpdateNickName	String	-	
UpdateProgram	String	-	
UpdateLevel	$\mathbb{N}$	-	
AddFriend	User	-	
DeleteFriend	User	IndexOutofBound Exception	
AddRequest	User	-	
DeleteRequest	User	IndexOutofBound Exception	
PinLecture	Lecture	-	
UnPinLecture	Lecture	IndexOutofBound Exception	
PinEvent	Event	-	
UnPinEvent	Event	IndexOutofBound Exception	

6.4 Semantics

6.4.1 State Variables

- User: User User of the account

6.4.2 Environment Variables

None

6.4.3 Assumptions

All the sate variables of User is accessible directly so there is no getters in the module.

6.4.4 Access Routine Semantics

UpdateNickName(newName):

- transition: $User.nickName := newName$
- output: none
- exception: none

UpdateProgram(newProgram):

- transition: $User.program := newProgram$
- output: none
- exception: none

UpdateLevel(newLevel):

- transition: $User.level := newLevel$
- output: none
- exception: none

AddFriend(newFriend):

- transition: $User.friends := User.friends + \{newFriend\}$
- output: none
- exception: none

DeleteFriend(targetFriend):

- transition: $User.friends := User.friends - \{targetFriend\}$
- output: none
- exception: $exc := targetFriend \notin User.friends \Rightarrow IndexOutofBoundException$

AddRequest(newFriend):

- transition: $User.friendRequests := User.friendRequests + \{newFriend\}$
- output: none
- exception: none

DeleteRequest(targetFriend):

- transition: $User.friendRequests := User.friendRequests - \{targetFriend\}$
- output: none
- exception: $exc := targetFriend \notin User.friendRequests \Rightarrow IndexOutofBoundException$

PinLecture(newLec):

- transition: $User.lectures := User.lectures + \{newLec\}$
- output: none
- exception: none

UnpinLecture(targetLec):

- transition: $User.friendRequests := User.lectures - \{targetLec\}$
- output: none
- exception: $exc := targetLec \notin User.lectures \Rightarrow IndexOutofBoundException$

PinEvent(newEvent):

- transition: $User.events := User.lectures + \{newEvent\}$
- output: none
- exception: none

UnpinLecture(targetEvent):

- transition: $User.friendRequests := User.events - \{targetEvent\}$
- output: none
- exception: $exc := targetEvent \notin User.events \Rightarrow IndexOutofBoundException$

6.4.5 Local Functions

None

7 MIS of Friend Manager Module

7.1 Module

FriendManager

7.2 Uses

Account Module, Chat Module, Unity Transform Type

7.3 Syntax

7.3.1 Exported Constants

None

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
DisplayFriendList	-	<Tranform>	-
onClickDeleteFriend	User	-	IndexOutofBound Exception
onClickViewFriend	User	-	IndexOutofBound Exception
onClickMessageFriend	User	2D seq of pixels	IndexOutofBound Exception
onClickSendRequest	User	$\mathbb{B}$	-

7.4 Semantics

7.4.1 State Variables

None

7.4.2 Environment Variables

None

7.4.3 Assumptions

Assume the singleton Account is accessible from this module.

7.4.4 Access Routine Semantics

DisplayFriendList():

- transition: none
- output: $out := friendContainer$ *where* $(\forall x : \mathbb{Z} | 0 \leq x \leq Account.friends.length :$ $friendsContainer[i].position, friendsContainer[i].content =$ $(0, i * HEIGHT), Account.friends[i]),$
- exception: none

onClickDeleteFriend(targetUser):

- transition: Account.DeleteFriend(targetUser)
- output: none
- exception: $exc := targetUser.email \notin Account.User.friends \Rightarrow IndexOutofBoundException$

onClickViewFriend(targetUser):

- transition: Switch scene to user profile where User = targetUser
- output: none
- exception: $exc := targetUser.email \notin Account.User.friends \Rightarrow IndexOutofBoundException$

onClickMessageFriend(targetUser):

- transition: Call Chat Module to establish a connection
- output: UI of friend chat between Account.User and targetUser
- exception: $exc := targetUser.email \notin Account.User.friends \Rightarrow IndexOutofBoundException$

onClickSendRequest(targetUser):

- transition: targetUser.AddRequest(Account1.User.email) if the current user has not send a request yet
- output: $Account1.User.email \notin targetUser.friendRequest$
- exception: none

7.4.5 Local Functions

None

7.4.6 Local Constants

HEIGHT = 300 px

8 MIS of Friend Request Module

8.1 Module

FriendRequest

8.2 Uses

Account Module, Unity Transform Type

8.3 Syntax

8.3.1 Exported Constants

None

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
DisplayRequestList	-	<Transform>	-
onClickAcceptRequest	User	-	IllegalArgument Exception
onClickIgnoreRequest	User	-	IllegalArgument Exception

8.4 Semantics

8.4.1 State Variables

None

8.4.2 Environment Variables

None

8.4.3 Assumptions

Assume the singleton Account is accessible from this module.

8.4.4 Access Routine Semantics

DisplayRequestList():

- transition: none

- output: $out := requestContainer$ *where* $(\forall x : \mathbb{Z} | 0 \leq x \leq Account.friendRequests.length :$

  $requestContainer[i].position, requestContainer[i].content =$
  $(0, i * HEIGHT), Account.friendRequests[i]),$

- exception: none

onClickAcceptRequest(targetUser):

- transition: targetUser.friends := targetUser.friends + Account.User.email
  Account.User.AddFriend(targetUser)
  $Account.User.DeleteRequest(targetUser)$

- output: none

- exception: $exc := targetUser \notin Account.User.friendRequests \Rightarrow$
  $IllegalArgumentException$

onClickIgnoreRequest(targetUser):

- transition: $Account.User.DeleteRequest(targetUser)$

- output: none

- exception: $exc := targetUser \notin Account.User.friendRequests \Rightarrow$
  $IllegalArgumentException$

8.4.5 Local Functions

UpdateBadge(): String
It returns the content of friend request badge given the request number

- transition: none

- output: $out := requestNum = 0 \Rightarrow emptystring$
  $0 < requestNum < 100 \Rightarrow requestNum$
  $100 \leq requestNum \Rightarrow 99+$

- exception: none

8.4.6 Local Constants

HEIGHT = 150 px

9 MIS of Activity Detail View Module

9.1 Module

ActivityDetailView

9.2 Uses

Database Module, Permission Module

9.3 Syntax

9.3.1 Exported Constants

None

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
ViewActivities	-	-	-
AddActivity	Activity	-	InvalidPermission Exception
EditActivity	Activity, Activity	-	IndexOutofBound Exception, InvalidPermission Exception
DeleteActivity	Activity	-	IndexOutofBound Exception, InvalidPermission Exception
PinActivity	Activity	-	-
UnpinActivity	Activity	-	IndexOutofBound Exception

9.4 Semantics

9.4.1 State Variables

- activities: set of Activity
- pinnedActivities: set of Activity

9.4.2 Environment Variables

None

9.4.3 Assumptions

Activity is a generic class with <T> and it can be instantiated with type Lecture and Event. The singleton module Permission is accessible from this module.

9.4.4 Access Routine Semantics

ViewActivities():

- transition: Display activities
- output: none
- exception: none

AddActivity(newActivity):

- transition: $activities := activities + \{newActivity\}$
- output: none
- exception: $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

EditActivity(targetActivity, editedActivity):

- transition: $activities := activities - \{targetActivity\} + \{editedActivity\}$
- output: none
- exception: $exc := targetActivity \notin activities \Rightarrow IndexOutofBoundException$, $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

DeleteActivity(targetActivity):

- transition: $activities := activities - \{targetActivity\}$
- output: none
- exception: $exc := targetActivity \notin activities \Rightarrow IndexOutofBoundException$, $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

PinActivity(newActivity):

- transition: $pinnedActivities := pinnedActivities + \{newActivity\}$
- output: none
- exception: none

UnpinActivity(targetActivity):

- transition: $pinnedActivities := pinnedActivities - \{targetActivity\}$
- output: none
- exception: $exc := targetActivity \notin pinnedActivities \Rightarrow IndexOutofBoundException$

9.4.5 Local Functions

None

9.4.6 Local Constants

None

10 MIS of Lecture Detail View Module

10.1 Module

LectureDetailView
Inherit Activity Detail View Module (Activity Detail View <Lecture>)

10.2 Uses

Activity Detail View Module, Lecture Module

10.3 Syntax

10.3.1 Exported Constants

None

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
ViewActivities	-	-	-
AddActivity	Lecture	-	InvalidPermission Exception
EditActivity	Lecture, Lecture	-	IndexOutofBound Exception, InvalidPermission Exception
DeleteActivity	Lecture	-	IndexOutofBound Exception, InvalidPermission Exception
PinActivity	Lecture	-	-
UnpinActivity	Lecture	-	IndexOutofBound Exception

10.4 Semantics

10.4.1 State Variables

- activities: set of Lecture
- pinnedActivities: set of Lecture

10.4.2 Environment Variables

None

10.4.3 Assumptions

The singleton module Permission is accessible from this module.

10.4.4 Access Routine Semantics

ViewActivities():

- transition: Display lectures
- output: none
- exception: none

AddActivity(newActivity):

- transition: $activities := activities + \{newActivity\}$
- output: none
- exception: $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

EditActivity(targetActivity, editedActivity):

- transition: $activities := activities - \{targetActivity\} + \{editedActivity\}$
- output: none
- exception: $exc := targetActivity \notin activities \Rightarrow IndexOutofBoundException$, $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

DeleteActivity(targetActivity):

- transition: $activities := activities - \{targetActivity\}$
- output: none
- exception: $exc := targetActivity \notin activities \Rightarrow IndexOutofBoundException$, $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

PinActivity(newActivity):

- transition: $pinnedActivities := pinnedActivities + \{newActivity\}$
- output: none
- exception: none

UnpinActivity(targetActivity):

- transition: $pinnedActivities := pinnedActivities - \{targetActivity\}$
- output: none
- exception: $exc := targetActivity \notin pinnedActivities \Rightarrow IndexOutofBoundException$

10.4.5 Local Functions

None

10.4.6 Local Constants

None

11 MIS of Event Detail View Module

11.1 Module

EventDetailView
Inherit Activity Detail View Module (Activity Detail View <Event>)

11.2 Uses

Activity Detail View Module, Event Module

11.3 Syntax

11.3.1 Exported Constants

None

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
ViewActivities	-	-	-
AddActivity	Event	-	InvalidPermission Exception
EditActivity	Event, Event	-	IndexOutofBound Exception, InvalidPermission Exception
DeleteActivity	Event	-	IndexOutofBound Exception, InvalidPermission Exception
PinActivity	Event	-	-
UnpinActivity	Event	-	IndexOutofBound Exception

11.4 Semantics

11.4.1 State Variables

- activities: set of Event
- pinnedActivities: set of Event

11.4.2 Environment Variables

None

11.4.3 Assumptions

The singleton module Permission is accessible from this module.

11.4.4 Access Routine Semantics

ViewActivities():

- transition: Display events
- output: none
- exception: none

AddActivity(newActivity):

- transition: $activities := activities + \{newActivity\}$
- output: none
- exception: $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

EditActivity(targetActivity, editedActivity):

- transition: $activities := activities - \{targetActivity\} + \{editedActivity\}$
- output: none
- exception: $exc := targetActivity \notin activities \Rightarrow IndexOutofBoundException$, $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

DeleteActivity(targetActivity):

- transition: $activities := activities - \{targetActivity\}$
- output: none
- exception: $exc := targetActivity \notin activities \Rightarrow IndexOutofBoundException$, $exc := \neg Permission.isAdmin \Rightarrow InvalidPermissionException$

PinActivity(newActivity):

- transition: $pinnedActivities := pinnedActivities + \{newActivity\}$
- output: none
- exception: none

UnpinActivity(targetActivity):

- transition: $pinnedActivities := pinnedActivities - \{targetActivity\}$
- output: none
- exception: $exc := targetActivity \notin pinnedActivities \Rightarrow IndexOutofBoundException$

11.4.5 Local Functions

None

11.4.6 Local Constants

None

12 MIS of Authentication Module

12.1 Module

Authentication

12.2 Uses

User Module, Database Module, Server Module

12.3 Syntax

12.3.1 Exported Constants

None

12.3.2 Exported Access Programs

12.4 Semantics

12.4.1 State Variables

- User: FirebaseUser

12.4.2 Environment Variables

None

12.4.3 Assumptions

The user will have a unique account and only has access to that account.

12.4.4 Access Routine Semantics

Register():

- transition: $friends := GetFriendsFromDB(currentUser)$
- output: none
- exception: none

12.4.5 Local Functions

Login(_email, _password):

- transition: $\exists < \_email, \_password > \in FirebaseAuth \Rightarrow Login$
- output: $User = AuthResult.CurrentUser$
- exception: $exc := \neg(\exists < \_email, \_password > \in FirebaseAuth) \Rightarrow AuthFailedException$

Register():

- transition: $\neg(\exists \_email \in FirebaseAuth) \rightarrow FirebaseAuth.add(User) \wedge FirebaseDatabase.add(User)$
- output: $User \in FirebaseAuth \wedge User \in FirebaseDatabase$
- exception: $\exists \_email \in FirebaseAuth \rightarrow IllegalDatabaseOperationException$

12.4.6 Local Constants

- auth: FirebaseAuth
- DatabaseReference: DatabaseReference

13 MIS of Database Module

13.1 Module

FirebaseDatabase
This module uses Firebase Realtime Database library. For details of all syntax and semantics of exported constants and access programs, see Firebase database documentation. documentation

13.2 Uses

13.3 Syntax

13.3.1 Exported Constants

See Firebase database documentation.

13.3.2 Exported Access Programs

The following table will show some functions the application uses most frequently, for more details, see Firebase database documentation.

Name	In	Out	Exceptions
Child	String	DatabaseReference	PermissionDenied, NetworkError, ExpiredToken
HasChild	String	$\mathbb{B}$	PermissionDenied, NetworkError, ExpiredToken
RemoveValueAsync	String	Task$<\mathbb{B}>$	PermissionDenied, NetworkError, ExpiredToken
SetValueAsync	String , String	Task$<\mathbb{B}>$	PermissionDenied, NetworkError, ExpiredToken
GetValueAsync	String	Task<DataSnapshot>	PermissionDenied, NetworkError, ExpiredToken
GoOffline	-	-	PermissionDenied, NetworkError, ExpiredToken
GoOnline	-	-	PermissionDenied, NetworkError, ExpiredToken

13.4 Semantics

13.4.1 State Variables

None

13.4.2 Environment Variables

- DBreference: Firebase.Database.DatabaseReference
  A reference to the root location of this database
- User: Firebase.Auth.FirebaseUser
  The current user that operates this database
- PermittedUsers: set of String
  The list of user emails that are allowed to read the database content
- Admins: set of String
  The list of user emails that are allowed to edit the database content

13.4.3 Assumptions

Assume the database connection is stable and it will not disconnect unless the user disconnect it manually.

13.4.4 Access Routine Semantics

Child(pathString):

- transition: none
- output: *out* := DatabaseReference to pathString relative to the root
- exception: $exc := NoInternet \Rightarrow NetworkError \mid TokenExpired \Rightarrow ExpiredToken \mid User.email \notin PermittedUsers \Rightarrow PermissionDenied$

HasChild(pathString):

- transition: none
- output: $out := DBreference.Child(pathString) =$ null
- exception: $exc := NoInternet \Rightarrow NetworkError \mid TokenExpired \Rightarrow ExpiredToken \mid User.email \notin PermittedUsers \Rightarrow PermissionDenied$

RemoveValueAsync(pathString):

- transition: $DBreference.Child(pathString) :=$ null
- output: $out := DBreference.HasChild(pathString)$
- exception: $exc := NoInternet \Rightarrow NetworkError \mid TokenExpired \Rightarrow ExpiredToken \mid User.email \notin Admins \Rightarrow PermissionDenied$

SetValueAsync(pathString, value):

- transition: $DBreference.Child(pathString) := value$
- output: $out := DBreference.Child(pathString) = value$
- exception: $exc := NoInternet \Rightarrow NetworkError \mid TokenExpired \Rightarrow ExpiredToken \mid User.email \notin Admins \Rightarrow PermissionDenied$

GetValueAsync(pathString):

- transition: none
- output: *out* := Snapshot of $DBreference.Child(pathString)$

- exception: $exc := NoInternet \Rightarrow NetworkError \mid TokenExpired \Rightarrow ExpiredToken \mid User.email \notin PermittedUsers \Rightarrow PermissionDenied$

GoOffline():

- transition: Manually disconnect the FirebaseDatabase client from the server and disable automatic reconnection.
- output: none
- exception: $exc := NoInternet \Rightarrow NetworkError \mid TokenExpired \Rightarrow ExpiredToken \mid User.email \notin Admins \Rightarrow PermissionDenied$

GoOnline():

- transition: Manually reestablish a connection to the FirebaseDatabase server and enable automatic reconnection.
- output: none
- exception: $exc := NoInternet \Rightarrow NetworkError \mid TokenExpired \Rightarrow ExpiredToken \mid User.email \notin Admins \Rightarrow PermissionDenied$

13.4.5 Local Functions

None

14 MIS of Server Module

14.1 Module

RTCServer

14.2 Uses

14.3 Syntax

14.3.1 Exported Constants

14.3.2 Exported Access Programs

Name	In	Out	Exceptions
SendMessage	User, String	Task	-
SendLocation	Group, Double, Double	Task	-

14.4 Semantics

14.4.1 State Variables

14.4.2 Environment Variables

14.4.3 Assumptions

User identifiers are unique.

14.4.4 Access Routine Semantics

SendMessage(recipient, msg):

- transition: none
- output: *out* := Task; *out*.IsCompleted := $True$
- exception: none

SendLocation(friendGroup, lat, lon):

- transition: none
- output: *out* := Task; *out*.IsCompleted := $True$
- exception: none

14.4.5 Local Functions

None

15 MIS of AR Camera

15.1 Module

AR Camera

15.2 Uses

15.3 Syntax

15.3.1 Exported Constants

15.3.2 Exported Access Programs

Name	In	Out	Exceptions
DetectTarget	-	-	-

15.4 Semantics

15.4.1 State Variables

15.4.2 Environment Variables

- cameraFeed: 2D array of pixels
- sceneCamera: Camera
- imageTargets: list of Target
- scanTargets: list of Target

15.4.3 Assumptions

15.4.4 Access Routine Semantics

DetectTarget():

- transition: Implicitly invokes the AR Interface when a valid target is detected.
- output: none
- exception: none

15.4.5 Local Functions

None

16 MIS of AR Interface

16.1 Module

AR Interface

16.2 Uses

16.3 Syntax

16.3.1 Exported Constants

16.3.2 Exported Access Programs

Name	In	Out	Exceptions
HandleInput	-	-	-
Display	-	-	-

16.4 Semantics

16.4.1 State Variables

16.4.2 Environment Variables

16.4.3 Assumptions

16.4.4 Access Routine Semantics

HandleInput():

- transition: none
- output: none
- exception: none

Display():

- transition: none
- output: none
- exception: none

16.4.5 Local Functions

None

17 MIS of MapBox

17.1 Module

MapBox

17.2 Uses

17.3 Syntax

17.3.1 Exported Constants

17.3.2 Exported Access Programs

Name	In	Out	Exceptions
Display	-	-	-

17.4 Semantics

17.4.1 State Variables

17.4.2 Environment Variables

- APIKey: String

17.4.3 Assumptions

17.4.4 Access Routine Semantics

Display():

- transition: none
- output: none
- exception: none

17.4.5 Local Functions

None

18 MIS of Map Interface

18.1 Module

Map Interface

18.2 Uses

18.3 Syntax

18.3.1 Exported Constants

18.3.2 Exported Access Programs

Name	In	Out	Exceptions
HandleInput	-	-	-

18.4 Semantics

18.4.1 State Variables

- building: list of BuildingLocation

18.4.2 Environment Variables

- camera: Camera

18.4.3 Assumptions

18.4.4 Access Routine Semantics

HandleInput():

- transition: Opens user interface when a building marker is tapped
- output: none
- exception: none

18.4.5 Local Functions

None

19 MIS of Friend Chat

19.1 Module

Friend Chat

19.2 Uses

19.3 Syntax

19.3.1 Exported Constants

19.3.2 Exported Access Programs

Name	In	Out	Exceptions
StartConnection	String, String	-	-
SendMessage	User, String	-	-
ReceiveMessage	User, String	-	-

19.4 Semantics

19.4.1 State Variables

- connection: HubConnection
- onMessageReceived: Action

19.4.2 Environment Variables

19.4.3 Assumptions

19.4.4 Access Routine Semantics

StartConnection(url, handler):

- transition: Creates a new HubConnection and stores it in connection. Connects the given handler to the server endpoint.
- output: none
- exception: none

SendMessage(recipient, message):

- transition: Sends a message to the recipient through the server hub connection.
- output: none

- exception: none

ReceiveMessage(sender, message):

- transition: Receives a message from the server hub connection. The sender's id is received as well.
- output: none
- exception: none

19.4.5 Local Functions

None

20 Appendix

20.1 Database Tables

User

Column Name	Type	Description
email	String	ID of a user
nickName	(Optional) String	Nickname/display name of a user
photoUri	(Optional) Uri	Visual Avatar
program	(Optional) String	Study field
level	(Optional) int	Level of program
friends	(Optional) <User>	List of friends
friendRequests	(Optional) <User>	List of requesters
lectures	(Optional) <Lecture>	List of pinned lecture
events	(Optional) <Event>	List of pinned event

Lecture

Column Name	Type	Description
code	String	ID of a course, course code
name	(Optional) String	formal name of a course
instructor	(Optional) String	name of the instructor
time	(Optional) String	Includes start and end time in a weekly schedule
location	(Optional) String	Building and room

Event

Column Name	Type	Description
name	String	ID of an event
description	(Optional) String	event description
organizer	(Optional) String	organizer of the event
startTime	(Optional) DateTime	when it starts
duration	(Optional) int	how long is the event (in minutes)
location	(Optional) String	Building and room
isPublic	$\mathbb{B}$	If it is a public event

References

Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.

Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL http://citeseer.ist.psu.edu/428727.html.