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 Lecture Module

6.1 Module

Lecture

6.2 Uses

None

6.3 Syntax

6.3.1 Exported Constants

None

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
Lecture	String, String, String, String, String	Lecture	-

6.4 Semantics

6.4.1 State Variables

- Code: Lecture code
- Name: Lecture name
- instructor: Lecture instructor
- time: Lecture time
- location: Lecture location

6.4.2 Environment Variables

None

6.4.3 Assumptions

Strings passed as input are of valid format.

6.4.4 Access Routine Semantics

Lecture(code, name, instructor, time, location):

- transition: Initializes an instance of the 'Lecture' class with the given parameters.
- output: Instance of 'Lecture'.
- exception: none

6.4.5 Local Functions

None

7 MIS of Event Module

7.1 Module

Event

7.2 Uses

None

7.3 Syntax

7.3.1 Exported Constants

None

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
Event	String, String, String, String	Event	-

7.4 Semantics

7.4.1 State Variables

- Name: Event name
- Date: Event date
- Location: Event location
- Organizer: Event organizer

7.4.2 Environment Variables

None

7.4.3 Assumptions

Parameters to the constructor are of the correct format.

7.4.4 Access Routine Semantics

Event(name, date, location, organizer):

- transition: Initializes an instance of the 'Event' class with the given parameters.
- output: Instance of 'Event'.
- exception: none

7.4.5 Local Functions

None

8 MIS of Account Module

8.1 Module

Account

8.2 Uses

Database Module, User Module, Authentication Module

8.3 Syntax

8.3.1 Exported Constants

None

8.3.2 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	

8.4 Semantics

8.4.1 State Variables

- User: User User of the account

8.4.2 Environment Variables

None

8.4.3 Assumptions

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

8.4.4 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: $exc := targetEvent \notin User.events \Rightarrow IndexOutofBoundException$

8.4.5 Local Functions

None

9 MIS of Friend Manager Module

9.1 Module

FriendManager

9.2 Uses

Account Module, Chat Module, Unity Transform Type

9.3 Syntax

9.3.1 Exported Constants

None

9.3.2 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	$\mathbb{B}$	-

9.4 Semantics

9.4.1 State Variables

None

9.4.2 Environment Variables

None

9.4.3 Assumptions

Assume the singleton Account is accessible from this module.

9.4.4 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

9.4.5 Local Functions

None

9.4.6 Local Constants

HEIGHT = 300 px

10 MIS of Friend Request Module

10.1 Module

FriendRequest

10.2 Uses

Account Module, Unity Transform Type

10.3 Syntax

10.3.1 Exported Constants

None

10.3.2 Exported Access Programs

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

10.4 Semantics

10.4.1 State Variables

None

10.4.2 Environment Variables

None

10.4.3 Assumptions

Assume the singleton Account is accessible from this module.

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

10.4.5 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

10.4.6 Local Constants

HEIGHT = 150 px

11 MIS of Activity Detail View Module

11.1 Module

ActivityDetailView

11.2 Uses

Database Module, Permission Module

11.3 Syntax

11.3.1 Exported Constants

None

11.3.2 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

11.4 Semantics

11.4.1 State Variables

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

11.4.2 Environment Variables

None

11.4.3 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.

11.4.4 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: $exc := targetActivity \notin pinnedActivities \Rightarrow IndexOutofBoundException$

11.4.5 Local Functions

None

11.4.6 Local Constants

None

12 MIS of Lecture Detail View Module

12.1 Module

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

12.2 Uses

Activity Detail View Module, Lecture Module

12.3 Syntax

12.3.1 Exported Constants

None

12.3.2 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

12.4 Semantics

12.4.1 State Variables

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

12.4.2 Environment Variables

None

12.4.3 Assumptions

The singleton module Permission is accessible from this module.

12.4.4 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: $exc := targetActivity \notin pinnedActivities \Rightarrow IndexOutofBoundException$

12.4.5 Local Functions

None

12.4.6 Local Constants

None

13 MIS of Event Detail View Module

13.1 Module

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

13.2 Uses

Activity Detail View Module, Event Module

13.3 Syntax

13.3.1 Exported Constants

None

13.3.2 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

13.4 Semantics

13.4.1 State Variables

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

13.4.2 Environment Variables

None

13.4.3 Assumptions

The singleton module Permission is accessible from this module.

13.4.4 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: $exc := targetActivity \notin pinnedActivities \Rightarrow IndexOutofBoundException$

13.4.5 Local Functions

None

13.4.6 Local Constants

None

14 MIS of Authentication Module

14.1 Module

Authentication

14.2 Uses

User Module, Database Module, Server Module

14.3 Syntax

14.3.1 Exported Constants

None

14.3.2 Exported Access Programs

14.4 Semantics

14.4.1 State Variables

- User: FirebaseUser

14.4.2 Environment Variables

None

14.4.3 Assumptions

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

14.4.4 Access Routine Semantics

Register():

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

14.4.5 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: $\exists \_email \in FirebaseAuth \rightarrow IllegalDatabaseOperationException$

14.4.6 Local Constants

- auth: FirebaseAuth
- DatabaseReference: DatabaseReference

15 MIS of Lecture List Manager Module

15.1 Module

Lecture List Manager

15.2 Uses

Lecture Module, Database Module

15.3 Syntax

15.3.1 Exported Constants

None

15.3.2 Exported Access Programs

Name	In	Out	Exceptions
Start	-	-	InitializationException
WriteNewLec	-	-	DatabaseException
GetLectures	Action¡string¿		DatabaseException
GetLectureData	-	-	CoroutineException
ExitDataPage	-	-	SceneLoadingException

15.4 Semantics

15.4.1 State Variables

- databaseReference
- lecList
- lecCode
- lecName
- lecInstructor

15.4.2 Environment Variables

None

15.4.3 Assumptions

The lecture data is in the correct format and the database is accessible.

15.4.4 Access Routine Semantics

Start():

- transition: Initializes the databaseReference and calls GetLectureData.
- output: none
- exception: InitializationException

WriteNewLec():

- transition: Creates a new Lecture object with the data from lecCode, lecName, and lecInstructor, and writes it to the Firebase database.
- output: none
- exception: DatabaseException

GetLectures(onCallBack):

- transition: Retrieves the lecture data from the Firebase database.
- output: none
- exception: DatabaseException

GetLectureData():

- transition: Calls GetLectures and sets the lecture data to the retrieved data.
- output: none
- exception: CoroutineException

ExitDataPage():

- transition: Changes the scene to "MenuScene".
- output: none
- exception: SceneLoadingException

15.4.5 Local Functions

None

16 MIS of Event List Manager Module

16.1 Module

Event List Manager

16.2 Uses

Event Module, Database Module

16.3 Syntax

16.3.1 Exported Constants

None

16.3.2 Exported Access Programs

Name	In	Out	Exceptions
Start	-	-	-
Update	-	-	-
HandleChildAdded	object, ChildChangedEventArgs	-	DatabaseError
WriteNewEvent	-	-	InvalidDateFormatException, DatabaseException
GetEvents	Action¡string¿	IEnumerator	-
GetEventData	-	-	-
ExitDataPage	-	-	-
OnDestroy	-	-	-

16.4 Semantics

16.4.1 State Variables

- databaseReference
- eventList
- scrollView
- eventsTextRectTransform

- errorMessage
- originalPosition
- eventName
- eventDate
- eventLocation
- eventOrganizer
- scrollRect

16.4.2 Environment Variables

- eventList
- scrollView
- eventsTextRectTransform
- errorMessage
- eventName
- eventDate
- eventLocation
- eventOrganizer
- scrollRect

16.4.3 Assumptions

The event data is in the correct format and the database is accessible.

16.4.4 Access Routine Semantics

Start():

- transition: Initializes the databaseReference, originalPosition, and subscribes to the ChildAdded event. Calls GetEventData.
- output: none
- exception: none

Update():

- transition: Updates the position of the eventsTextRectTransform based on the active state of scrollView.
- output: none
- exception: none

HandleChildAdded(object sender, ChildChangedEventArgs args):

- transition: If there is no database error, calls GetEvents to update the event list.
- output: none
- exception: DatabaseError

WriteNewEvent():

- transition: Validates the event date format, creates a new Event object, and writes it to the database.
- output: none
- exception: InvalidDateFormatException, DatabaseException

GetEvents(Action¡string¿ onCallBack):

- transition: Retrieves event data from the database and formats it into a string.
- output: IEnumerator
- exception: none

GetEventData():

- transition: Calls GetEvents to retrieve event data and updates the eventList text and scrollRect position.
- output: none
- exception: none

ExitDataPage():

- transition: Loads the "LoginScene".
- output: none
- exception: none

OnDestroy():

- transition: Unsubscribes from the ChildAdded event.
- output: none
- exception: none

16.4.5 Local Functions

None

17 MIS of Database Module

17.1 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

17.2 Uses

17.3 Syntax

17.3.1 Exported Constants

See Firebase database documentation.

17.3.2 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

17.4 Semantics

17.4.1 State Variables

None

17.4.2 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

17.4.3 Assumptions

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

17.4.4 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: $exc := NoInternet \Rightarrow NetworkError \mid TokenExpired \Rightarrow ExpiredToken \mid User.email \notin Admins \Rightarrow PermissionDenied$

17.4.5 Local Functions

None

18 MIS of Server Module

18.1 Module

RTCServer

18.2 Uses

18.3 Syntax

18.3.1 Exported Constants

18.3.2 Exported Access Programs

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

18.4 Semantics

18.4.1 State Variables

18.4.2 Environment Variables

18.4.3 Assumptions

User identifiers are unique.

18.4.4 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

18.4.5 Local Functions

None

19 MIS of AR Camera

19.1 Module

AR Camera

19.2 Uses

19.3 Syntax

19.3.1 Exported Constants

19.3.2 Exported Access Programs

Name	In	Out	Exceptions
DetectTarget	-	-	-

19.4 Semantics

19.4.1 State Variables

19.4.2 Environment Variables

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

19.4.3 Assumptions

19.4.4 Access Routine Semantics

DetectTarget():

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

19.4.5 Local Functions

None

20 MIS of AR Interface

20.1 Module

AR Interface

20.2 Uses

20.3 Syntax

20.3.1 Exported Constants

20.3.2 Exported Access Programs

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

20.4 Semantics

20.4.1 State Variables

20.4.2 Environment Variables

20.4.3 Assumptions

20.4.4 Access Routine Semantics

HandleInput():

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

Display():

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

20.4.5 Local Functions

None

21 MIS of MapBox

21.1 Module

MapBox

21.2 Uses

21.3 Syntax

21.3.1 Exported Constants

21.3.2 Exported Access Programs

Name	In	Out	Exceptions
Display	-	-	-

21.4 Semantics

21.4.1 State Variables

21.4.2 Environment Variables

- APIKey: String

21.4.3 Assumptions

21.4.4 Access Routine Semantics

Display():

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

21.4.5 Local Functions

None

22 MIS of Map Interface

22.1 Module

Map Interface

22.2 Uses

22.3 Syntax

22.3.1 Exported Constants

22.3.2 Exported Access Programs

Name	In	Out	Exceptions
HandleInput	-	-	-

22.4 Semantics

22.4.1 State Variables

- building: list of BuildingLocation

22.4.2 Environment Variables

- camera: Camera

22.4.3 Assumptions

22.4.4 Access Routine Semantics

HandleInput():

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

22.4.5 Local Functions

None

23 MIS of Friend Chat

23.1 Module

Friend Chat

23.2 Uses

23.3 Syntax

23.3.1 Exported Constants

23.3.2 Exported Access Programs

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

23.4 Semantics

23.4.1 State Variables

- connection: HubConnection
- onMessageReceived: Action

23.4.2 Environment Variables

23.4.3 Assumptions

23.4.4 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

23.4.5 Local Functions

None

24 Appendix

24.1 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.