SE 3XA3: Module Interface Specification
SocialPy

Team 1, SocialPy
Anando Zaman, zamana11
Graeme Woods, woodsg1
Yuvraj Randhawa, randhawy
Due March 5, 2021
Tag DD-Rev.1

April 8, 2021

Contents

List of Tables

List of Figures

Table 1: Revision History

Date	Developer(s)	Change
March 13, 2021	Anando Zaman	Copy template
March 14, 2021	Anando Zaman	Made progress on Profile modules
March 14, 2021	Anando Zaman	Completed MIS for Authentication and command parser modules
March 17, 2021	Anando Zaman	Finalized CommandParser, Authentication, firebase_creds, ViewPost, Add Followers, DeleteAccount and Query modules.
March 17, 2021	Yuvraj Randhawa	Completed DeletePost, EditName, & EditLocation modules
March 17, 2021	Graeme Woods	Completed DeletePost, EditPost, and EditLocation modules

CommandParser Module

Module

Uses

ViewPost
AddPost
EditPost
DeletePost
AddFollowers
DeleteFollowers
ViewFollowers

Brief Description

This class represents a the Command Parser. It details the suite of functions that are able to parse/execute commands. It is therefore a controller module.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
parseCommand()	command: str	GUI	Invalid Command alert
post_commands()	command_action: str, command_context: str	GUI	Invalid Command alert
profile_commands()	command_action: str, command_context: str	GUI	Invalid Command alert

0.1 Interface Semantics

0.1.1 State Variables

running: Boolean value that controls whole program should exit/run.
firebase: firebase object instance to execute commands via Pyrebase API.
db: firebase database object instance used to execute database commands
user_token: string value that is used to identify an active authenticated user in order to execute commands to the database.

0.1.2 Environmental Variables

Display Device: Screen Input: Keyboard

0.1.3 State Invariant

None

0.1.4 Assumptions

parseCommand() is called before *post_commands*() & *profile_commands*()

0.1.5 Access Program Semantics

parseCommand(self,command)

- Transition: transition the user to a different module.
- output: success message and the action completed
- exception : if command is not in the set of available commands, raise invalid command exception.

post_commands(self,command_action, command_context)

- Transition: transition the user to add, view, delete
- Output: Success message that an action has been completed
- exception : if command is not in the set of available commands, raise invalid command exception.

profile_commands(self,command_action, command_context)

- Transition: transition the user to add, view, delete
- Output: Success message that an action has been completed
- exception : if command is not in the set of available commands, raise invalid command exception.

Authentication Module

Module

Uses

firebase_creds

Brief Description

This class represents all that is needed to Authenticate the user. It takes a Firebase instance and db instance for the constructor and executes the functions outlined below in order to authenticate or reset account credentials.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
login()	Input: str	GUI	Invalid credentials alert
register()	Input: str	GUI	Registration_Fail alert, Account_exists alert
password_reset()	Input: str	GUI	Invalid_email alert

0.2 Interface Semantics

0.2.1 State Variables

firebase: firebase object instance to execute commands via Pyrebase API.
db: firebase database object instance used to execute database commands

0.2.2 Environmental Variables

Display Device: Screen
Input: Keyboard

0.2.3 State Invariant

None

0.2.4 Assumptions

Valid internet connection established

0.2.5 Access Program Semantics

login()

- transition: Authenticates the user credentials with that found in the Firebase console. If successful, transitions to homescreen.

- output: displays success message.
- exception : If credentials pair is invalid or does not exist, raise Invalid Credentials alert.

register()

- transition: Creates an account for the user by uupdating the database with the account credentials and general information(name, city, username). If successful, transitions to homescreen.
- output: displays success message and homescreen.
- exception : If credentials pair is invalid or already exists, raise Account_exists alert. If registration failed for any other reason, raise Registration_Fail alert.

Password_reset()

- transition: Sends a password reset request to Firebase upon which Firebase sends an email to the user prompting to reset password.
- output: displays success message and homescreen.
- exception : If email is invalid or already exists, raise invalid email alert.

firebase_creds Module

Module

Uses

None

Brief Description

This class is used for initializing the Firebase API connection and contains methods for interaction. It contains the API setup along with the various getter/setter methods to make changes to the DB during first-start.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
get_db()	None	db: db_object	Session Unavailable Alert
sign_in()	email: str, password: str	None	Invalid Credentials Alert
sign_up()	email: str, password: str	None	Insufficient Email Alert
get_user_instance()	None	user: str	Session Unavailable Alert
get_username()	None	username: str	Session Unavailable Alert
set_username()	username: str	None	Session Unavailable Alert
get_UID()	None	UID: str	Session Unavailable Alert

0.3 Interface Semantics

0.3.1 State Variables

auth = firebase.auth object
user = str
username = str
db = database object

0.3.2 Environmental Variables

Input: Keyboard
Display Device: Screen

0.3.3 State Invariant

0.3.4 Assumptions

Valid internet connection exists in order to authenticate.

0.3.5 Access Program Semantics

get_db()

- transition: None
- output: self.db
- exception : If user is not yet logged in, then a database connection has not yet been established and thus Session Unavailable Alert occurs.

sign_in(username, password)

- transition: self.username := username input. self.db := self.firebase.db
- output: Success message to the GUI
- exception : If user is not yet logged in, then a database connection has not yet been established and thus Invalid Credentials Alert occurs.

sign_up(username, password)

- transition: self.username := username input. self.db := self.firebase.db
- output: Success message to the GUI
- exception : If user email is already registered or invalid, then Insufficient Email Alert is raised.

get_user_instance()

- transition: None
- output: self.user
- exception : If user credentials are insufficient, then a Session Unavailable Alert occurs.

get_username()

- transition: None
- output: self.username
- exception : If user credentials are insufficient, then a Session Unavailable Alert occurs.

set_username(username)

- transition: self.username := username
- output: Success message to the GUI

- exception : If user credentials are insufficient, then a Session Unavailable Alert occurs.

get_UID()

- transition: None
- output: self.auth.get_account_info(self.user['idToken'])["users"][0]['localId']
- exception : If user credentials are insufficient, then a Session Unavailable Alert occurs.

AddPost Module

Module

Uses

firebase_creds

Brief Description

Used for addingPosts by the specified user.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
addPost()	db: DB_Object, username: str, content: str, user_token: str	post: str	None

0.4 Interface Semantics

0.4.1 State Variables

None

0.4.2 Environmental Variables

Input: Keyboard
Display Device: Screen

0.4.3 State Invariant

None

0.4.4 Assumptions

User is authenticated

0.4.5 Access Program Semantics

addPost(db, username, content, user_token)

- transition: Uploads an instance of a post to the database under the currently authenticated user.
- output Success message to the GUI
- exception : None

DeletePost Module

Module

Uses

firebase_creds

Brief Description

Used for Deleting a post that exists and corresponds to a user.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
DeletePost()	db: DB_Object, current_user: str, user_token: str, post_id: str	GUI	InvalidPostID alert

0.5 Interface Semantics

0.5.1 State Variables

None

0.5.2 Environmental Variables

Input: Keyboard
Display Device: Screen

0.5.3 State Invariant

None

0.5.4 Assumptions

User is authenticated

0.5.5 Access Program Semantics

DeletePost(db, current_user, user_token, post_id)

- transition: Deletes a specified post from the database under the currently authenticated user.
- output := Success message to the GUI
- exception : If inputted post id does not exist, raise InvalidPostID alert.

EditPost Module

Module

Uses

firebase_creds

Brief Description

Used for Editing an existing post that corresponds to the user.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
EditPost()	db: db_object, current_user: str, user_token: str, post_id: str, content: str	GUI	-

0.6 Interface Semantics

0.6.1 State Variables

0.6.2 Environmental Variables

Input: Keyboard
Display Device: Screen

0.6.3 State Invariant

None

0.6.4 Assumptions

User is authenticated

0.6.5 Access Program Semantics

EditPost(db, current_user, user_token, post_id, content)

- transition: Edits an existing post corresponding to the given post_id in the database.
- output Success message to the GUI
- exception : None

QueryPost Module

Module

Uses

firebase_creds

Brief Description

Used for Querying existing posts that correspond to a specific user or for querying ALL posts available in the database.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
QueryPostsByUser()	db: db_object, username: str, user_token: str	postContent: list[String]	Invalid_user alert
QueryPostsAll()	db: db_object, user_token: str	postContent: list[String]	None
QueryPostsFollowing()	db: db_object, firebase: firebase_object user_token: str	postContent: HashMap[String]	Invalid_user alert

0.7 Interface Semantics

0.7.1 State Variables

None

0.7.2 Environmental Variables

Input: Keyboard
Display: Screen

0.7.3 State Invariant

None

0.7.4 Assumptions

Assumed that the user is logged in.

0.7.5 Access Program Semantics

QueryPostsByUser(db, username, user_token)

- transition: None
- output: Returns a list containing all the posts of a user
- exception : If user does not exist, raise Invalid_user alert

0.7.6 Access Program Semantics

QueryPostsAll(db, user_token)

- transition: None
- output: Returns a list of ALL the posts
- exception : None

0.7.7 Access Program Semantics

QueryPostsFollowing(db, username, user_token)

- transition: None
- output: Returns a dictionary containing the posts of all users you follow.
- exception : If user does not exist, raise Invalid_user alert

ViewPost Module

Module

Uses

firebase_creds

Brief Description

Used for Viewing/displaying existing posts to the screen that correspond to a specific user or for querying ALL posts available in the database.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
ViewPostsByUser()	db: db_object, username: str, user_token: str	GUI	InvalidUser Alert
ViewPostsAll()	db: db_object, user_token: str	GUI	InvalidUser Alert
ViewPostsFollowing()	db: db_object, firebase: firebase_object user_token: str	GUI	InvalidUser Alert

0.8 Interface Semantics

0.8.1 State Variables

None

0.8.2 Environmental Variables

Input: Keyboard
Display: Screen

0.8.3 State Invariant

None

0.8.4 Assumptions

User is logged in, thus a valid user session exists.

0.8.5 Access Program Semantics

ViewPostsByUser()

- transition:
- output := displays the posts information by a user to the screen.
- exception : If user does not exist, raise InvalidUser alert

ViewPostsAll()

- transition: None
- output := displays ALL the posts information to the screen.
- exception : If user does not exist, raise InvalidUser alert

ViewPostsFollowing()

- transition: None
- output := displays the posts information based on the followings of the user, to the screen.
- exception : If user does not exist, raise InvalidUser alert

AddFollowers Module

Module

Uses

firebase_creds

Brief Description

Used for adding users to the current logged in users' followings list.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
AddFollowers()	db: db_object, username: str, user_to_follow: str, user_token: str	GUI	InvalidUser Alert

0.9 Interface Semantics

0.9.1 State Variables

None

0.9.2 Environmental Variables

Input: Keyboard
Display: Screen

0.9.3 State Invariant

None

0.9.4 Assumptions

Already logged in

0.9.5 Access Program Semantics

AddFollowers(db, current_user, user_to_follow, user_token)

- transition: Appends the user_to_follow to the current_users' followers list
- output := success message to the screen
- exception : If user does not exist, raise InvalidUser Alert.

DeleteAccount Module

Module

Uses

firebase_creds

Brief Description

Used for deleting the logged in users' account.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
DeleteAccount()	firebase: firebase_object, db: db_object, username: str, current_user: str, user_token: str	GUI	None

0.10 Interface Semantics

0.10.1 State Variables

None

0.10.2 Environmental Variables

Input: Keyboard
Display: Screen

0.10.3 State Invariant

None

0.10.4 Assumptions

User already logged in

0.10.5 Access Program Semantics

DeleteAccount()

- transition: Removes the account from the firebase DB along with any posts associated with the user from the AllPosts branch and profile branches
- output := Success message to the screen.
- exception : None

DeleteFollowers Module

Module

Uses

firebase_creds

Brief Description

Used for deleting the followers for a given user.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
DeleteFollowers()	db: db_object, username: str, current_user: str, user_to_remove: str, user_token: str	GUI	InvalidUser Alert

0.11 Interface Semantics

0.11.1 State Variables

None

0.11.2 Environmental Variables

Input: Keyboard
Display: Screen

0.11.3 State Invariant

None

0.11.4 Assumptions

Already logged in

0.11.5 Access Program Semantics

DeleteFollowers(db, current_user, user_to_remove, user_token)

- transition: Remove the user_to_remove from the current_users' followings list.
- output := Success message to the screen.
- exception : If user to remove does not exist in the current_users' following list, then raise InvalidUser Alert.

EditLocation Module

Module

Uses

firebase_creds

Brief Description

Used for updating the location of the logged in users' profile.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
EditLocation()	db: db_object, username: str, new_location: str, user_token: str	GUI	InvalidLocation Alert

0.12 Interface Semantics

0.12.1 State Variables

None

0.12.2 Environmental Variables

Input: Keyboard
Display: Screen

0.12.3 State Invariant

None

0.12.4 Assumptions

User already logged in

0.12.5 Access Program Semantics

EditLocation(db, username, user_token, new_location)

- transition: Updates location of the currently authenticated user in the database.
- output := Success message to the GUI
- exception : If inputted location is empty, raise InvalidLocation Alert

EditName Module

Module

Uses

firebase_creds

Brief Description

Used for updating the name information of the current logged in users' profile.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
EditName()	db: db_object, username: str, new_name: str, user_token: str	GUI	InvalidName Alert

0.13 Interface Semantics

0.13.1 State Variables

0.13.2 Environmental Variables

Input: Keyboard
Display: Screen

0.13.3 State Invariant

0.13.4 Assumptions

0.13.5 Access Program Semantics

EditName(db, username, user_token, new_name)

- transition: Updates name of the currently authenticated user in the database.
- output := Success message to the GUI
- exception : If inputted name is empty, raise InvalidName Alert

QueryFollowers Module

Module

Uses

firebase_creds

Brief Description

Used for Querying existing the followers list for a given user.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
QueryFollowing()	db: db_object, username: str, user_token: str	List[String]	InvalidUser Alert
QueryFollowers()	db: db_object, username: str, user_token: str	List[String]	InvalidUser Alert

0.14 Interface Semantics

0.14.1 State Variables

0.14.2 Environmental Variables

Input: Keyboard
Display: Screen

0.14.3 State Invariant

None

0.14.4 Assumptions

User already logged in.

0.14.5 Access Program Semantics

QueryFollowing(db, username,user_token)

- transition: None
- output := Returns the followings list of a user containing the usernames that they follow.
- exception : If user does not exist or is invalid, raise InvalidUser Alert.

QueryFollowers(db, username,user_token)

- transition:
- output := Returns the followers list of the user containing the usernames that follow them.
- exception : If user does not exist or is invalid, raise InvalidUser Alert.

QueryProfile Module

Module

Uses

firebase_creds

Brief Description

Used for Querying the profile of a given user.

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
QueryProfile()	db: db_object, username: str, user_token: str	Hashmap[String: String]	InvalidUser Alert

0.15 Interface Semantics

0.15.1 State Variables

None

0.15.2 Environmental Variables

Input: Keyboard
Display: Screen

0.15.3 State Invariant

None

0.15.4 Assumptions

User already logged in.

0.15.5 Access Program Semantics

QueryProfile(db, username, user_token)

- transition: None
- output := Returns a Python Dictionary (Hashmap) of the profile information(posts, location, name) for the given username.
- exception : If username is invalid or does not exist, raise InvalidUser Alert.

ViewFollowers Module

Module

Uses

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
ViewFollowers()	db: db_object, username: str, user_token: str	GUI	UnknownUsername, NoFollowers
ViewFollowing()	db: db_object, username: str, user_token: str	GUI	UnknownUsername, NoFollowing

0.16 Interface Semantics

0.16.1 State Variables

0.16.2 Environmental Variables

Input: Keyboard
Display: Screen

0.16.3 State Invariant

None

0.16.4 Assumptions

User already logged in.

0.16.5 Access Program Semantics

ViewFollowers(db, username,user_token)

- transition: None
- output: Follower list for given user
- exception: If there is no user with given username raise UnknownUsername & if there is no followers for the given user raise NoFollowers exception.

ViewFollowing(db, username,user_token)

- transition: None
- output: Following list for given user
- exception: If there is no user with given username raise UnknownUsername & if the given user does not follow anyone raise NoFollowing exception.

ViewProfile Module

Module

Uses

Interface Syntax

Exported Constants

None

Exported Types

None

Exported Access Programs

Routine Name	In	Out	Exceptions
ViewProfile()	db: db_object, username: str, user_token: str	GUI	UnknownUsername

0.17 Interface Semantics

0.17.1 State Variables

0.17.2 Environmental Variables

Input: Keyboard
Display: Screen

0.17.3 State Invariant

None

0.17.4 Assumptions

User already logged in.

0.17.5 Access Program Semantics

ViewProfile(db, username,user_token)

- transition: None
- output: Profile information of the given user.
- exception: If there is no user with given username raise UnknownUsername.