Rita Samir
ID : 25

Sohayla Mohammed
ID : 31

Merit Vector
ID : 69

Youmna Dwidar
ID : 73

CSED GP Library: Milestone 4

11th May 2020

OVERVIEW

We proposed a web application represented as a library of resources for all previous graduation projects and for the upcoming graduation projects, students can upload their codes and their documentation. We also propose the web application as an open environment for communication and sharing experience between all computer science and engineering students.

In this milestone we started the development of the second Agile increment.

Requirements Elicitation

I. Requirements Specification

We used Structured specifications to illustrate our requirements :

Function	User's Verification.
Inputs	List of the users to be verified. Each user has information : Name, email, Departement, Graduation year, Position, National ID, Phone number.
Outputs	Decision of the teaching staff (approve/disapprove). A suitable email to be sent to the user based on that decision.
Action	Show pending users to the admin and take their decision about the user; whether verify or not, and send an email to the user with the appropriate action depending on the admin's decision.
Requirements	- A valid admin user is already authenticated. - Admin user is logged in.
Pre-conditions	- User has registered. - Administrative email set-up.
Post-conditions	- if the user is verified, send them an email with the verification link. - If the post is not verified, remove this user from the database after

	sending an email to them.
Side effects	Remove user from database if not verified.

Function	Add followers to posts
Inputs	The post that needs to be followed and the users that follow this post(who comment on this post and the citations of this post).
Outputs	In case of comment: a notification will be sent to all the followers.
Action	At first when a post is added to the list of posts, all the citations of this post will be added as a followers to this post. Also when a user writes a comment on this post a notification will be sent to all the followers and also this user will be added as a follower to this post.
Requirements	- A valid user is already authenticated. - The users (that adds the comment) are logged in.
Pre-conditions	The user is a comment writer or a one of the citations of the post.
Post-conditions	All users will be added to the database(the followers table)
Side effects	none.

Function	edit posts
Inputs	The id of the post that needs to be edited
Outputs	The post with the new fields showed
Action	Post is showed with all its old fields to the user. Edit what needed t be edited then submit the changes by updating the post.
Requirements	- A valid user is already authenticated. - A post that already been created
Pre-conditions	The user is one of the teaching staff cited, the post is approved or not. The user is the post writer if the post is unapproved or disapproved
Post-conditions	The new post fields are updated in the database.
Side effects	none.

Function	edit profile
Inputs	The id of the user with the info that need to be edited.
Outputs	The new profile info showed on the profile page.
Action	The fields that can be updated is shown to the user so he*she can edit them. Only the password, name, email and phone number can be edited, as the others need verification from the admin.
Requirements	- A valid user is already authenticated. - The user is logged in.
Pre-conditions	The user logged in is the profile owner.
Post-conditions	The new info for the user is updated in the database.
Side effects	none.

Function	Disapprove posts
Inputs	The id of the post to be disapproved
Outputs	The post is set as disapproved
Action	The teaching staff list the pending posts to approve or not. Disapproving the post sets the value of disapprove to be 1 and shown to the post owner as a disapproved post. Users can either delete the post or edit it.
Requirements	- A valid user is already authenticated. - The user is logged in.
Pre-conditions	The user logged in is a teaching staff
Post-conditions	The post is set to be disapproved in the database. Disappear from the pending posts page and appear in the writer disapproved list. A notification is sent.
Side effects	The post maybe deleted later by its owner.

Requirement validation by reviews :

Review checks :

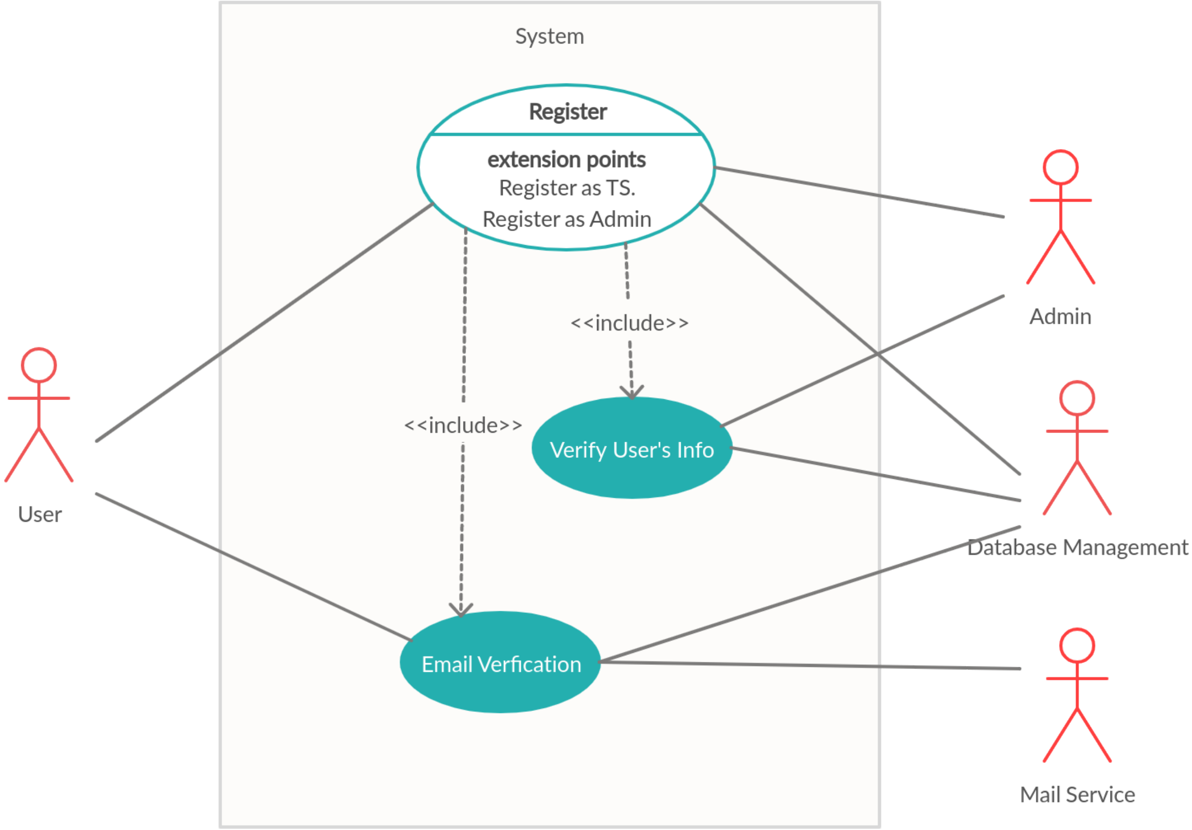
Verifiability	Comprehensibility	Traceability	Adaptability
✔	✔	✔	✔

Requirements validation by test-case generation :

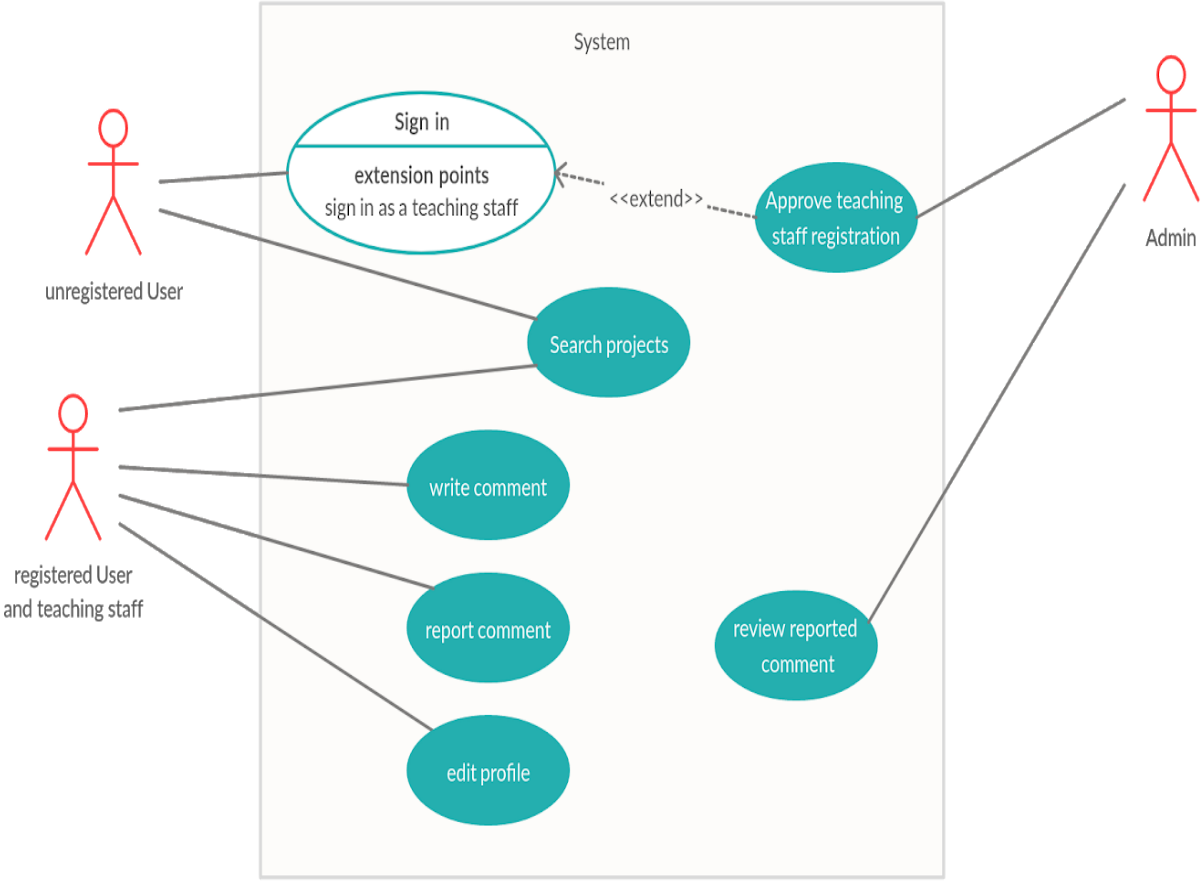
We did requirement validations by creating test cases containing those requirements and running those tests on the current model.

II. Use case diagrams

A. Registration

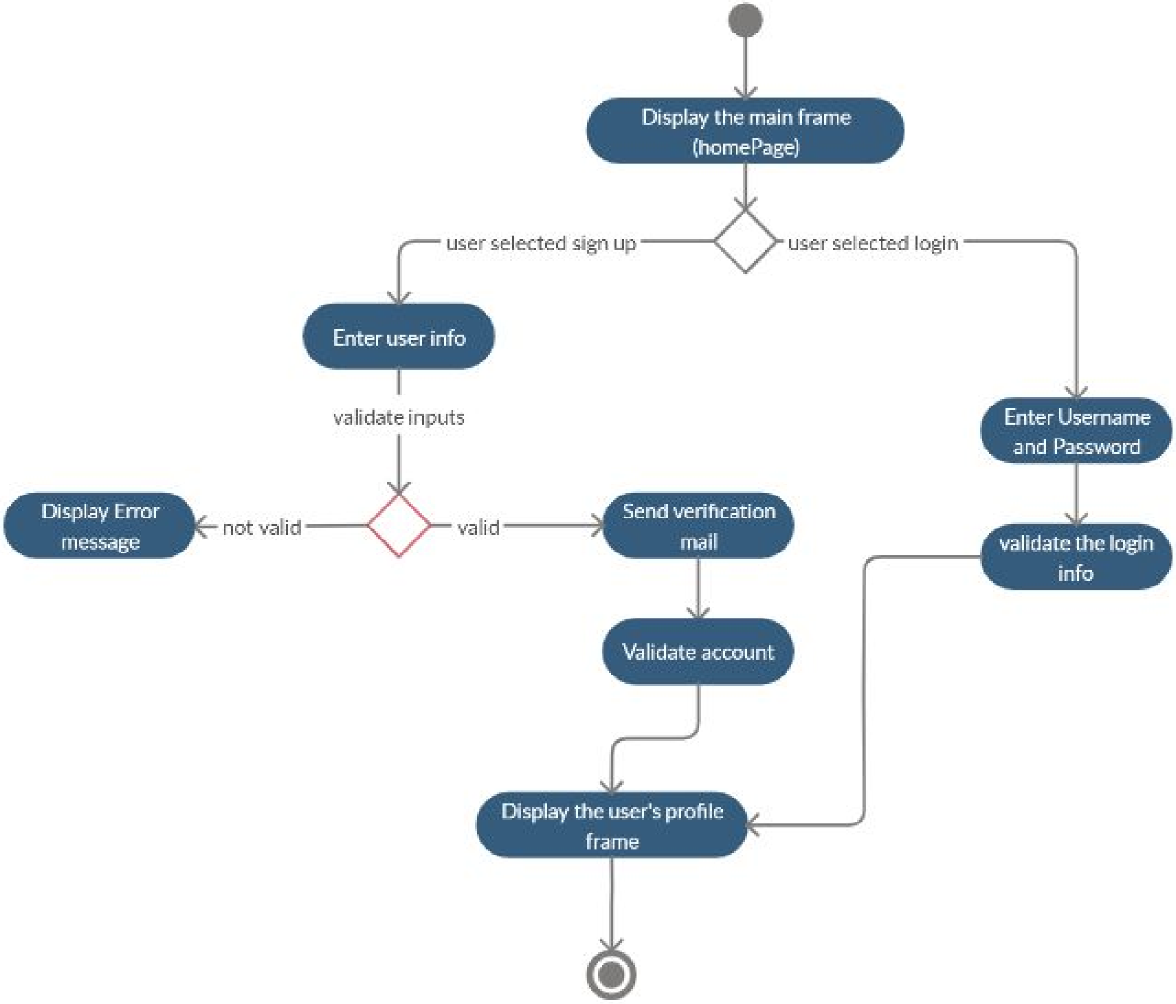


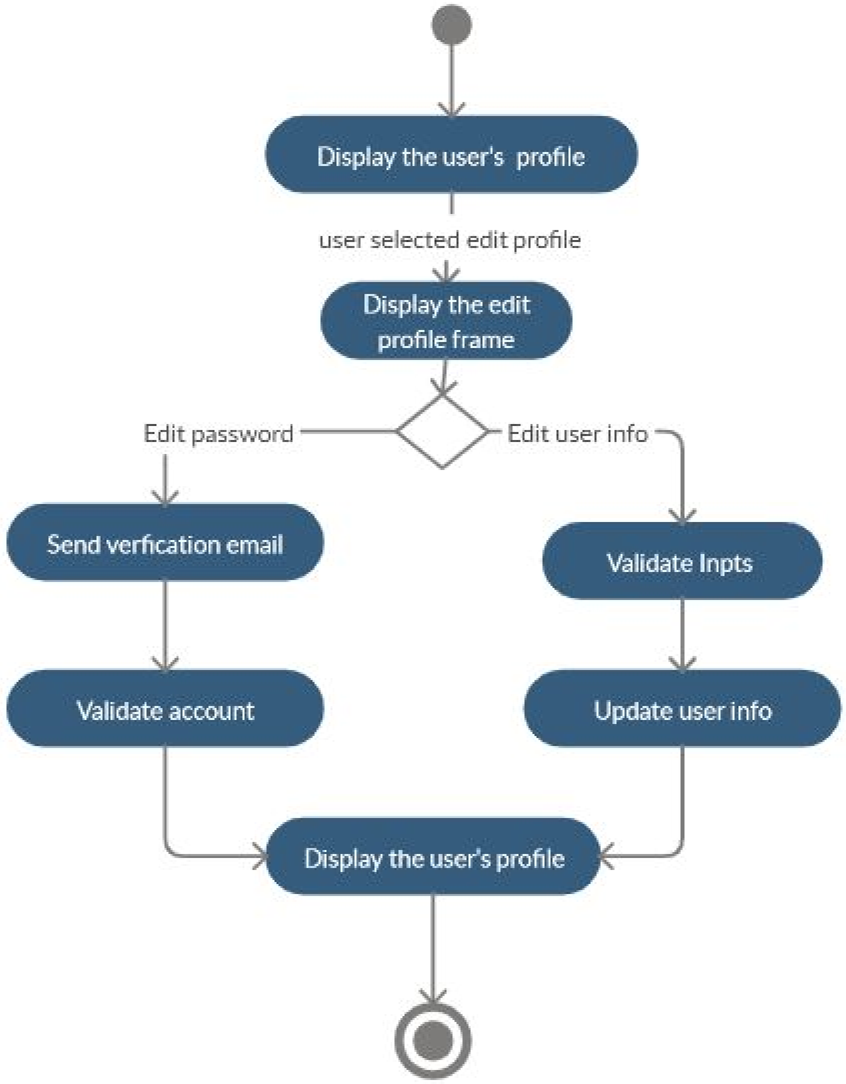
B. Edit Posts

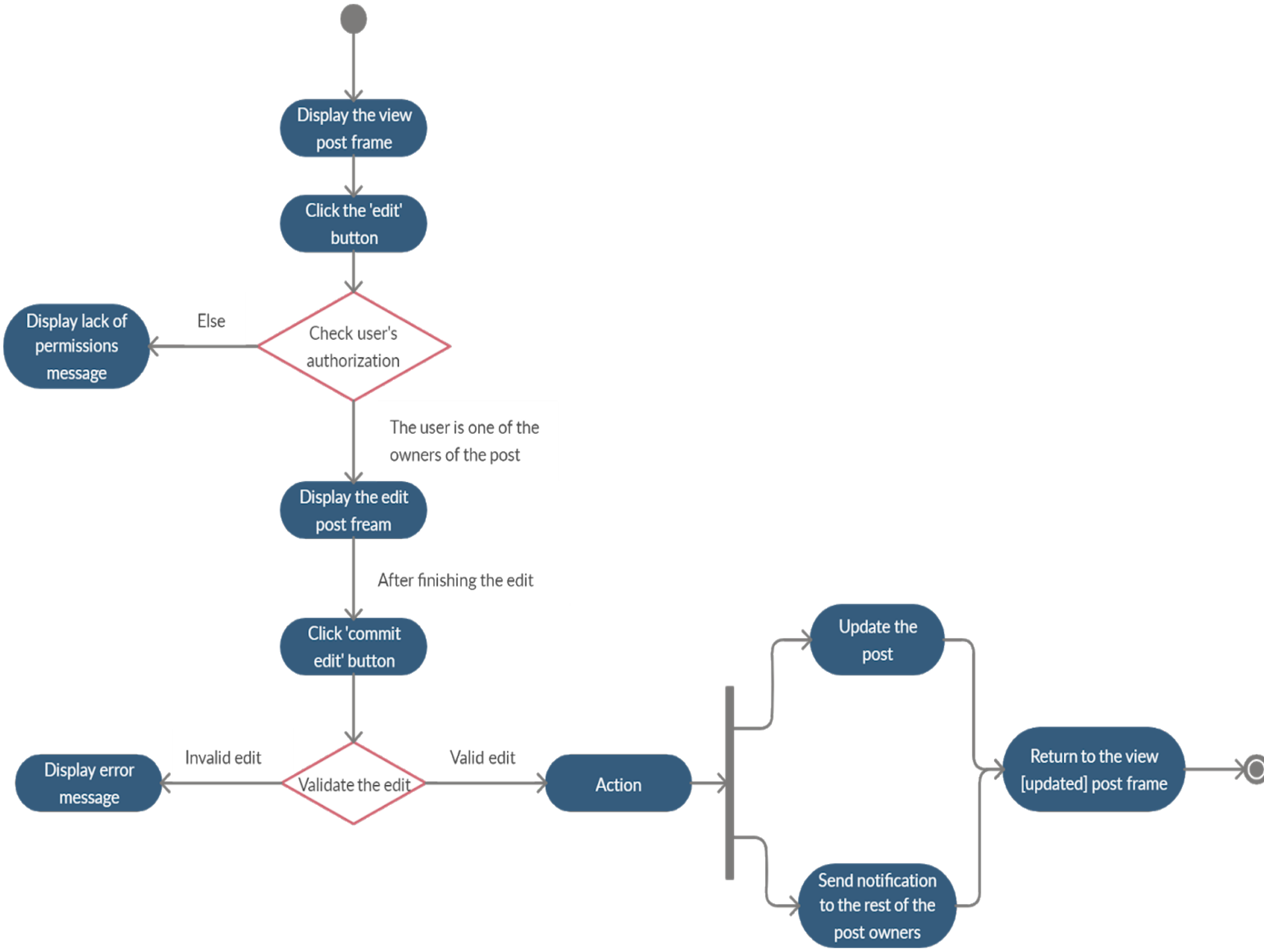


III. Activity diagrams

A. Verify User


Display the main frame (homePage)
user selected sign up
user selected login
Enter user info
validate inputs
Enter Username and Password
Display Error message
not valid
valid
Send verification mail
validate the login info
Validate account
Display the user's profile frame


Display the user's profile
user selected edit profile
Display the edit profile frame
Edit password
Edit user info
Send verfication email
Validate Inpts
Validate account
Update user info
Display the user's profile


Display the view post frame
Click the 'edit' button
Check user's authorization
Else
Display lack of permissions message
The user is one of the owners of the post
Display the edit post fream
After finishing the edit
Click 'commit edit' button
Validate the edit
Invalid edit
Display error message
Valid edit
Action
Update the post
Send notification to the rest of the post owners
Return to the view [updated] post frame

IV. Test cases

Admin functionality

Test	Functionality	Input	Output
Visitors can’t visit URLs	Ensure that visitors can’t visit admin reserved urls.	Get requests to admin reserved urls.	Redirecting to home page
Users type default can’t visit URLs	Ensure that Users type default can’t visit admin reserved urls.	Get requests to admin reserved urls.	Redirecting to home page
Users type admin can visit URLs	Ensure that Users type admin can visit admin reserved urls.	Get requests to admin reserved urls.	Assert a 200 status code.
Correct number of pending users shown	Ensure that the admin sees the correct number of pending users.	Get request to users/pending. Select request to database.	Assert that the correct page is visited and that the correct number of users has been retrieved from the database.
Unverified user can’t login	Ensure that the user can only login if they’re verified.	Post request to login url.	Assert redirect status (302).
Verify user info by admin results in the correct confirmation code.	Ensure that when the admin verifies a user a correct confirmation code is added, and that the user is removed from the pending list.	Post request to verify url with a specific user id.	Assert confirmation code is correct for the user. Assert number of pending users is valid.
Verify user’s email verification url.	Ensure that when a user follows the verification url, they are verified.	Get request to email verification url.	Assert redirect url to login page. Assert user becomes verified.
Not verifying user info	Ensure that when	Post request to	Assert user is removed

by admin results in the correct action.	the admin does not verify a user, the user is removed from database and the pending list.	un-verify url with a specific user id.	from the database. Assert redirection to pending users page. Assert user is removed from the pending list.

Following posts

	Functionality	Input	Output
TS adding new post	Ensure that citations of this post are added to the followers table.	TS creates a post with one student collaborator.	Followers table has 2 entries of the post for 2 citations.
TS approving new post	Ensure that citations of this post are added to the followers table after approving the post.	Student creates a post with one Ts collaborator and Ts approves that post.	Followers table has 2 entries of the post for 2 citations.
Adding comment	Ensure that the student that writes a comment will be added to the followers table.	A student writes a comment to an approved post that has 2 collaborators.	Followers table has 3 entries of the post for 2 citations and 1 comment writer.
Duplicate entries	Ensure that the followers table will ignore duplicate entries.	A student writes two comments to an approved post that has 2 collaborators.	Followers table has 3 entries of the post for 2 citations and 1 comment writer.
Adding comment Notifications	Ensure that the notifications have been sent to all users except the one who writes the comment.	A student writes a comment to an approved post that has 2 collaborators.	Notifications table has entries to the two citations only.

Edit Posts

	Functionality	Input	Output
Edit post fields	Ensure that the database is updated with the new fields	Send a put request with the modifications	The post is updated and redirected to its main page.
Edit post remove collaborators	Ensure that the citations table is updated correctly with the new citations	Send a put request with a smaller list of collaborators	The citations table removes the removed citations
Edit post change citations	Ensure that the citations table is updated correctly with the new citations	Send a put request with a different list of collaborators with old and new citations	The citations table removes the difference citations and add only the new ones
Edit post no edits	Ensure that the post didn’t change in the database	Send a put request with the post old fields unchanged	The post stays unchanged

Edit Profiles

	Functionality	Input	Output
Edit user info	Ensure that the database is updated with the new info	Send a put request with the modifications	The user info is updated and redirected to profile page
Edit user no info	Ensure that an error occurs if the data sent is not enough	Send a put request with insufficient changes	An error occurred