Team:	**Team Aguacate**
Test Procedure:	**Colon**
Date:	**6-Nov-13**
Performed by:	**Nelian Colon**
Reviewed by:	**Daniel Santiago**

Tools to be Used
Terminal
Sendgrid
Text editor (Sublime Text)
Chai
Mocha
Browser
Gitlab server
Node.js

Testing
To run all the tests go to home/panda and run sudo npm test. For all tests beginning with G, an instance of Gitlab, gitlab, is required. For the tests beginning with S, an instance of Sendgrid, sendgrid is required.

Test ID	Test Description	Instructions	Parameters	Expected Outcome	Pass/Fail	Comments
GP_1	Create Project	Call gitlab.project.create(params, callback)	params = {user_id: 2, name: 'happyhour'}	StatusCode: 201, body must contain information of the newly created project	✓	project id will be stored and used in later cases
GP_2	Fail Creating Project	Call gitlab.project.create(params, callback)	params = {user_id: 2, name: 'happyhour'}	Status Code: 404	✓	
GP_3	Get Projects	Call gitlab.project.getAll(callback)	-	Status Code: 200	✓	
GP_4	Get Project	Call gitlab.project.get(params, callback)	params = {id:projectId}	Status Code: 200, body must contain desired project information	✓	
GP_5	Get Inexistent Project	Call gitlab.project.get(params, callback)	params = {id:0}	Status Code: 404	✓	There is no project with id 0
GP_6	Populate Project (Initialize Repository)	Call gitlab.project.populate(params, callback)	params = {username: 'danysantiago', name: params.name, archive: __dirname + '/../res/+params.name+'.tar.gz'}	Error must not exist	✓	
GP_7	Add Member	Call gitlab.project.addMember(params, callback)	params = {id: projectId, user_id: 3}	Status Code: 201, body must contain recently added user information	✓	projectId is the id of the project created in GP_1
GP_8	Delete Project	Call gitlab.project.delete(params, callback)	params = {name: 'happyhour'}	Status Code is not 404	✓	
GP_9	Delete Inexistent Project	Call gitlab.project.delete(params,callback)	params = {name: 'happyhour'}	Status Code: 404	✓	

GR_1	Get Archive File	Call gitlab.repository.archive(params, out, callback)	with params={username: 'danysantiago', name: 'pandajavasnippets'}, out=fs.createWriteStream(__dirname+'/../res/'+params.name+'tar.gz');	Status Code: 200, pandajavasnippets.tar.gz is in panda/res/	✓	
GR_2	Get Branches	Call gitlab.repository.getBranches(params, callback)	params={id: projectId}	Status Code: 200	✓	
GR_3	Get Branch	Call gitlab.repository.getBranch(params, callback)	params={id:2, branch:'master'}	Status Code: 200, body must contain desired branch information	✓	
GR_4	Get Inexistent Branch	Call gitlab.repository.getBranch(params, callback)	params={id:2, branch:'bla'}	Status Code: 404	✓	there is no branch with name bla
GR_5	Get Commits	Call gitlab.repository.getCommits(params, callback)	params={id:2}	Status Code: 200	✓	
GR_6	Get Commit	Call gitlab.repository.getCommit(params, callback)	params={id:2, sha:'1ba94ed11'}	Status Code: 200, body must contain information of Initial Commit	✓	
GR_7	Get Diff	Call gitlab.repository.getDiff(params, callback)	params={id:2, sha:'1ba94ed11'}	Status Code: 200	✓	
GR_8	List Repository Tree	Call gitlab.repository.listTree(params, callback)	params={id:2}	Status Code: 200	✓	
GR_9	Get Raw Blob	Call gitlab.repository.getBlob(params, callback)	params={id:2, sha:'1ba94ed11', filepath: 'src/BadExampleProgram.java'}	Status Code: 200, body must contain source code for BadExampleProgram.java	✓	
GS_1	Login	Call gitlab.session.login(params, callback)	params={login: 'nelii28o2', password: 'mofongo'}	Status Code: 201, body must contain current user information (private token, etc)	✓	
GU_1	Create User	Call gitlab.user.create(fakeUser, callback)	fakeUser={email:'super@fake.com', password:'superfake', username: 'superfake', name: 'Fake'}	Status Code: 201, body must contain recently added user information	✓	user id will be stored and use it in other tests
GU_2	Fail Creating User	Call gitlab.user.create(fakeUser, callback)	fakeUser={email:'super@fake.com', password:'superfake', username: 'superfake', name: 'Fake'}	Status Code: 404	✓	
GU_3	Get Users	Call gitlab.user.getAll(callback)	-	Status Code: 404	✓	
GU_4	Get User	Call gitlab.user.get(params, callback)	params = {id:userId}	res.statusCode must be 200, body.id must equal params.id and body.name must equal fakeUser.name	✓	userId is the id of the user created in GU_1

GU_5	Modify User	Call gitlab.user.modify(params, callback)	params = {id: userId, email: 'other@email.com'}	res.statusCode must equal 200, body.email must not equal fakeUser.email, body.name must equal fakeUser.name	✓	
GU_6	Delete User	Call gitlab.user.delete(params, callback)	params = {id: userId}	res.statusCode must be 200, body.id must equal fakeUser.id	✓	
GU_7	Fail Getting already deleted user	Call gitlab.user.delete(params, callback)	params = {id: userId}	Status Code: 404	✓	
S_1	Send Email	Call sendgrid.sendEmail(params, callback)	params={from: 'no-reply@pandacode.sytes.net', to: 'nelian.colon@upr.edu', subject: 'Hello World', text: 'Email sent for happy hour'}	Error must not exist, Message must be success, Email is sent to nelian.colon@upr.edu	✓	