

General Summary of Integrative Software Engineering (10141)

What went well throughout the course and what things you will use in next courses?

We performed many meetings and through each meeting we finished a lot of stories from scrum board. At the beginning of every sprint we performed disclosure about what need to be done through spring.

What should be improved for next courses?

During some sprints we didn't properly understood sprint targets from Eyal for next courses additional checks for sprint's/course's targets should be performed. Synchronization and understanding of team member missions still may work better than it worked.

What was most enjoyable thing of this course

It was one of the hardest programming courses through all BSc but it was also course that we learned most and got a lot of tools from software engineering field. It was really great to use that many different technologies in the project and to see it works at the end with all this complex functionality. It was also funny to bypass 900 commits in github.

What would you do different in this course

We think defining requirement document through first course was redundant. We never used it because at the beginning of every sprint we got new targets and started understood better what need to be done. At first sprint we still lacked understanding of what need to be done so requirement document was redundant.

It would've save us time and headache if we started to work with constants earlier.

It may be better to show some code example of what we need to achieve at the end of the course.

We would like to see more comments with explanation in code. Sometimes it is hard to learn from code.

It would've been better to receive one last sprint remarks one week before we finish current sprint.

Done 1

Train on presentation at class room

Write pagination Gherkin for url #8

add pagination test for url #9

implement get all messages in message board

Move some constants to be configurable in properties file

Make adding message board an element operation

return exceptions instead of null in methods

add more steps to the setup document like unzipping files, where do the jars come from?

Make Question an element

Add gherkins of all activities

Change tests for url 10

+ Add another card

Done 2

In constants, near each function address add description on function name in comments

As a manager I want to be able to add questions to the session.

Move exceptions to relevant packages, and not concentrated in one place

Make adding question an element operation

Add guest role to an unverified user

playground of elements and user is decided by the server and not the user

add gherkin for url #10 with pagination

Only Player can use url#11 (Activity uri)

add test for url #10 with pagination

remove aspect in the controller

למן דור בספרייה ביום ראשון 13.1.2019 13:10

+ Add another card

Done 3

change RestfulAPI to RESTful web service using Spring Web

add sorting for data taken from database

Make tests simple, do not create users in the given if they are not needed for the test

remove redundant annotations from jpa activity

add explanations about element types in document

Client move to constants

we can rely on what the save method returns in Crud instead of using findById

indent all classes in code (ctrl+shift+f) - end of sprint

Make all system messages constant, for easier translation (supportability non functional requirement)

+ Add another card

Done 4

make video about project

add new tests to activity

Write pagination Gherkin for url #9

add comments to constants

add methods to DAO

add dates to all kanban boards, in addition to the sprint dates

remove unused properties

go over all the TODO in the project

Make sure we can search element by attribute name or attribute value(function 10)

synchronize requirements document and code

Improve workflows

+ Add another card

Done 5

make sure the tests are traceable - tests are the same as they are written in the gherkins (use same mails etc etc - look at sprint review of eyal)

create question database (dummy int)

Performance - access speed, storage usage, how many users can work at peak time or at average

Make sure we use readOnly in methods that we read from the database

Reliability - All security aspects in the app

Change url 10 to search by name and type

Add log to all methods in services

check if we need to change hashCode

add response element (for user answer), ask another group how to respond to user

synchronize requirements document

+ Add another card

Done 6

Remove playground field from Element Constructor

check if every class in client needs ClientModel model

use naming conventions, use capital letters for classes

write client. GUI

את המסמך שלכם במסמכים בטיחות

Fix constants names to be understandable

Add Goals for every use case

add pagination test for url #8

add keys of messages in message board attributes

As a student I want to know how to play the game

synchronize requirements document and code

+ Add another card

Done

Add gherkins of all activities

Change tests for url 10

Insert points logic

Check tests on document

Creating question via function 5 instead of 11

remove test of adding a duplicate element (because the service assigns the id)

creating messageboard via function 5 instead of 11

Add in services a method that receives a json string to create a new user/entity/activity in addition to the existing constructor in UserEntity/ActivityEntity/ElementEntity

remove transient from toString

add @Transient to setSuperkey()

+ Add another card

Done 2

13.1.2019 למנון חדר בספרייה ביום ראשון 13 נד 100

add getScore to activity

Move constants from Constants class to Playground Constants and refactor

in the appendix of used technologies, change from spring.io to the modules of spring that we are using

add gherkin element function 6.4 and 6.5

change constains for location

Move constants from Constants class to Activity Constants and refactor

Move constants from Constants class to User Constants and refactor

User cannot be able to update email and playground

+ Add another card

Done 3

Requirements document should explain what elements and what activities our system supports (user stories)

Every NON FUNCTIONAL requirement will include serial number, short description, and the letter S,U,R,P. Be moderate, not too much

write summary of the project

update numbering of questions in gherkin document

Create table of contents for the gherkin document

Change documentation according to project changes (look at first sprint highlights)

Add e-mail sending functionality for verification codes

Change strings into constants in newUserForm

+ Add another card

Done 4

add in jpa tests, tests that make sure entities are correctly saved in the database - do not rely on what the api returned, use service to make sure it's been saved

synchronize requirements document and code

add table of Contents for sprint reports

Remove suppress warning from project

Make equals method be dependent on the superkey only

Go over warnings

move createKey to service instead of using a static method in ActivityEntity

Do not use rest api to add user in the given, use service (find the test)

Write functions for GUI

+ Add another card

Done 5

write private on each method in client

delete code that is irrelevant and in comments - end of sprint

make verification code random

move createKey to service instead of using a static method in ElementEntity

new Font("TimesRoman", Font.BOLD, 20) make constants

Performance - access speed, storage usage, how many users can work at peak time or at average

Create summary for the course document

Add kanbans of all sprints in the summary of project

Remove from the interface methods that are use to inject services (autowires)

+ Add another card

Done 6

improve pagination for data pulls

Decide one or two non-functional requirements to focus in our code

Add nonfunctional requirements

As a student I want to be able to see how many points I have. (highscore)

Remove unused autowires in classes

we can only return activity if it's type is echo

remove sysout/syserr from code end of sprint

Make setters and getters of the same variable be adjacent to eachother

Traceability - make sure everything in the paperwork is implemented in the code and vice versa

write sprint report

+ Add another card

Gherkins