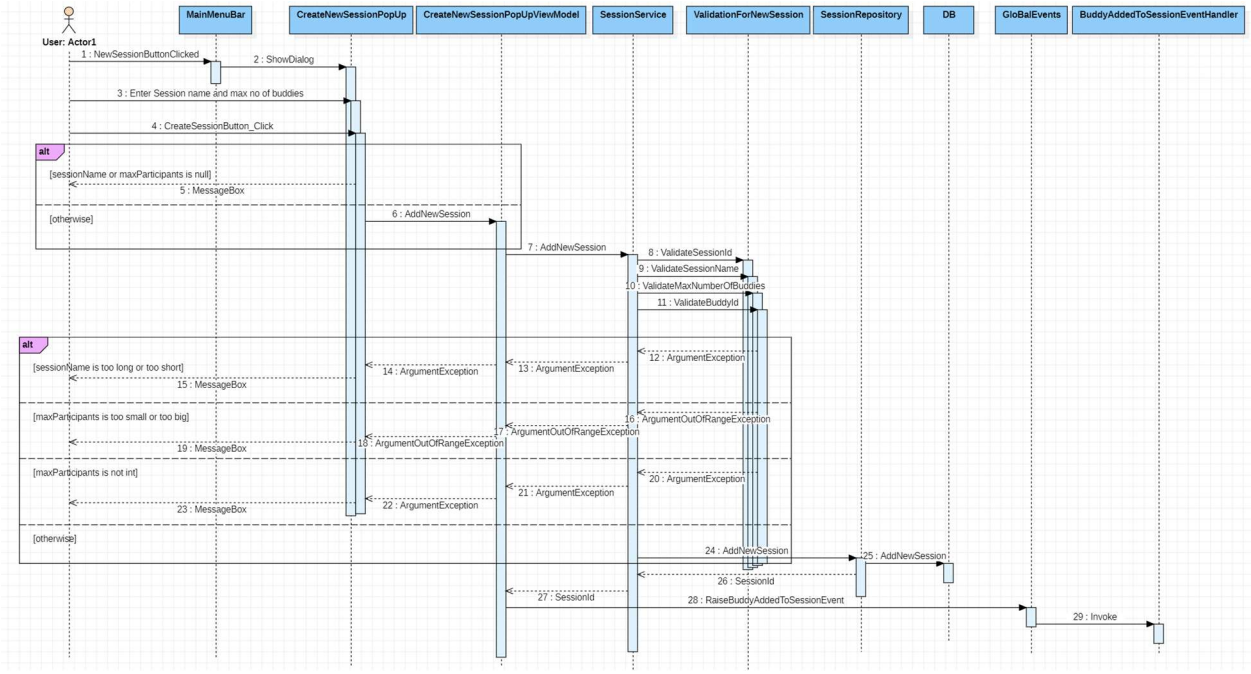
Diagrame ISS

Sequence diagram



https://youtu.be/pCK6prSq8aw?si=o3TQo4FpbW4PxGR7

Video de cam 9 minute unde mi se pare că explică bine ce e sequence diagram-ul și elementele folosite, în caz că explicațiile mele nu sunt prea clare.

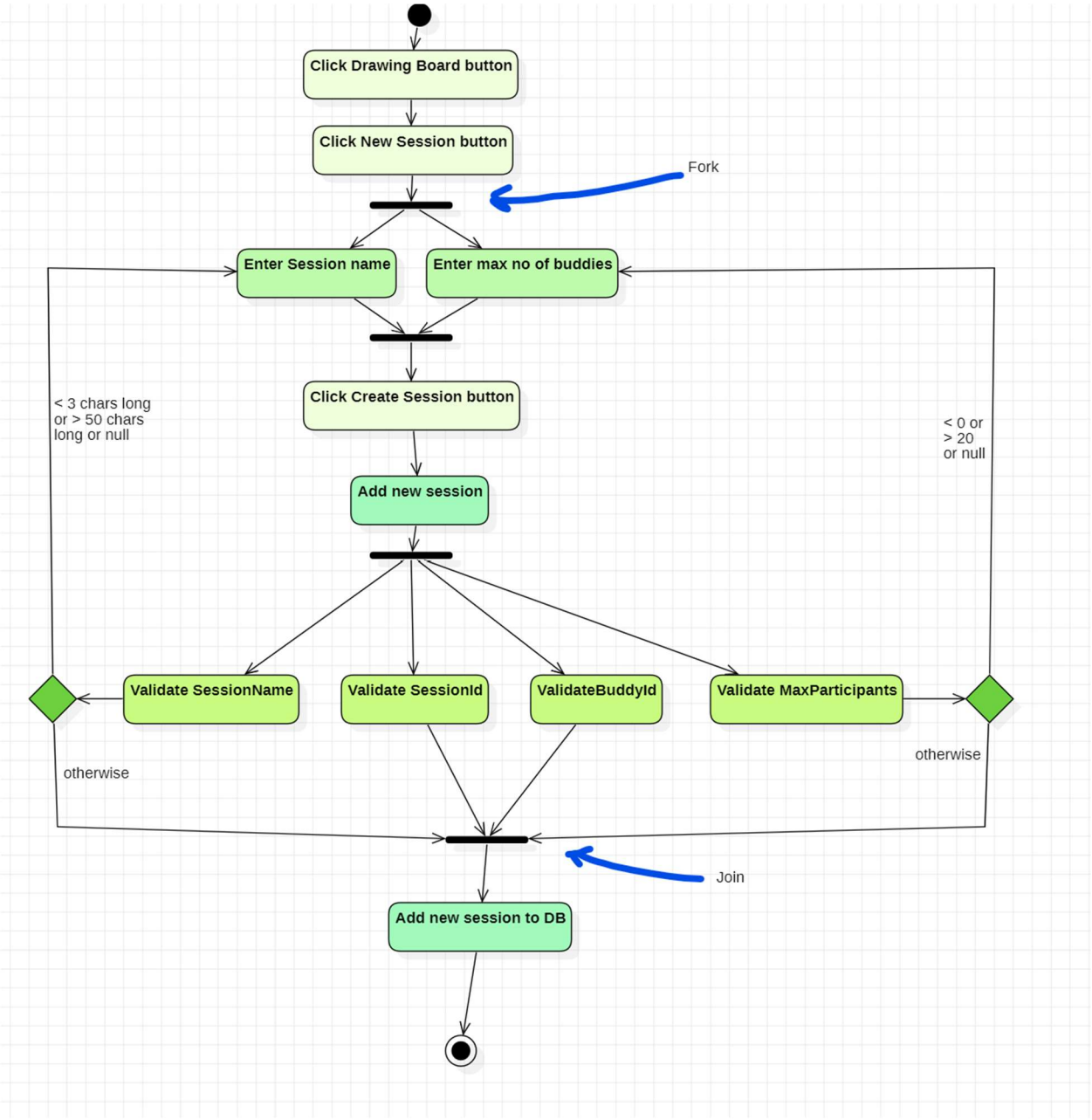
Important (chestii la care Imre a fost atent):

- Sus în căsuțele albastre neapărat să fie numele claselor exact așa cum sunt
- Pe săgeți să fie exact numele funcțiilor care se apelează, ce am uitat aici eu să pun sunt parametri la funcții !!!
- Săgețile să fie puse unele sub altele să se vadă exact ordinea evenimentelor (aici sunt și numerotate by default de aplicație, nu cred că trebuie să facem și noi asta)
- alt (alternative frame-ul) cum a fost explicat și în videoclip sunt ca un if else, condiția se pune între [ ]
- Barele acelea lungi albastre deschis ne arată până când durează apelul unei funcții, atât timp cât funcția e activă bara trebuie să existe (fără săgeți trase în gol!!!)
- Când se dă return la ceva neapărat cu săgeată punctată și scris pe ea ce returnează

Diagrama asta a fost creată pentru funcționalitatea create new session și se poate vedea foarte frumos ce se întâmplă, user-ul dă click pe un buton, îi apare o fereastră în care pune numele session-ului și numărul maxim de buddies din session și apasă pe

butonul de create, după care vine alternative frame-ul în care dacă una din cele două chestii e null va primi un message box, altfel se va apela AddNewSession unde se vor valida parametri și din nou depinde de caz, se returnează excepție dacă ceva nu e valid, dacă totul e ok se adaugă în baza de date.
Tot ce trebuie făcut este să mergeți pe cod și să puneți în diagramă în ordine ce și cum se apelează și ce se returnează.

Activity/Flow diagram



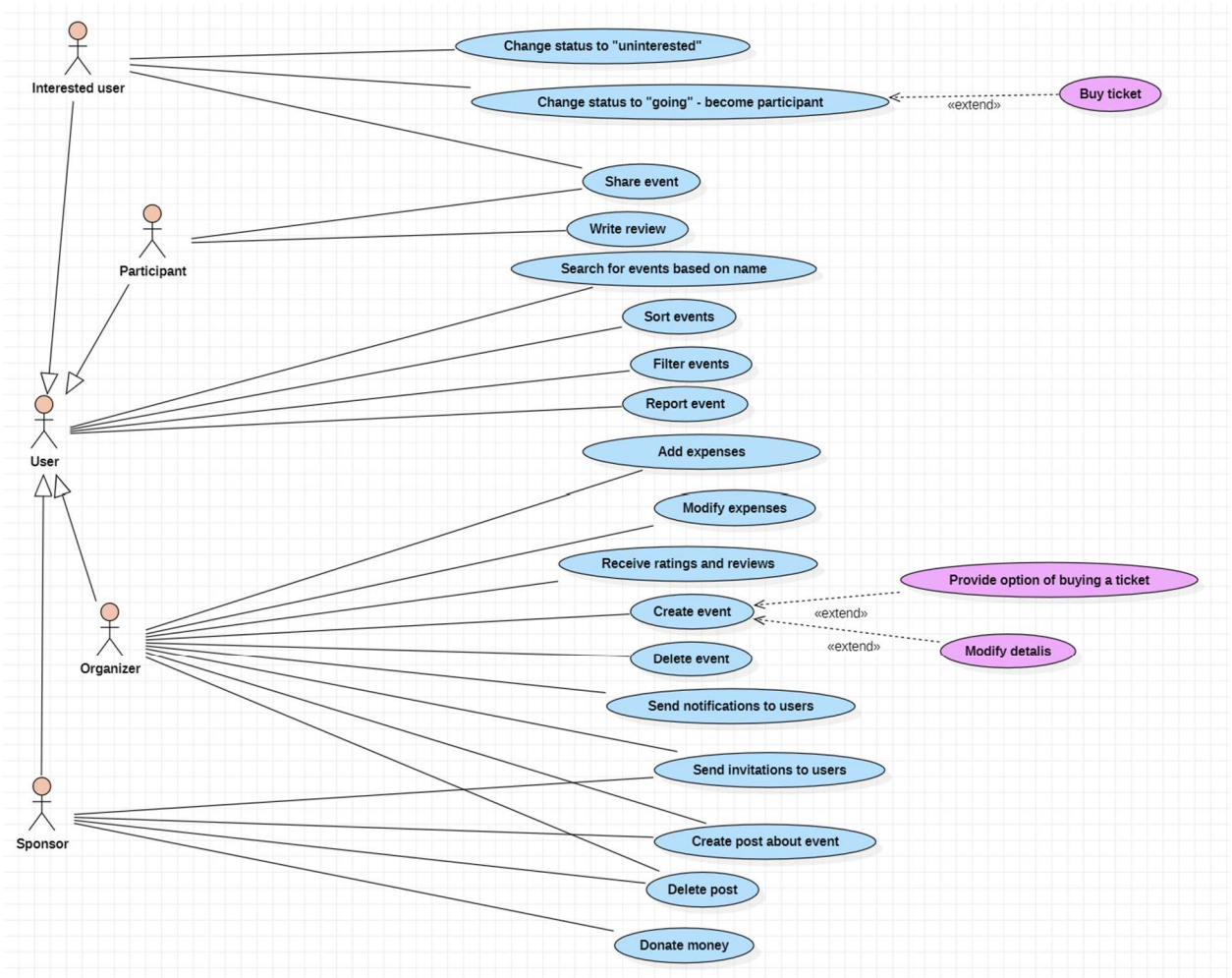
https://youtu.be/Iz6FnvhQ4Ms?si=0IxmKPmazC8Z3hf2

Video de cam 8 minute unde explică această diagramă.

Am făcut pentru aceeași funcționalitate ca și înainte, e cam același lucru doar că mai simplu reprezentat.

Avem un start point, un endpoint la final, activitățile în dreptunghiuri, fork unde activitățile se fac cam simultan să zicem și după se unesc toate înapoi la join. Rombul e ca și alternative-ul de mai devreme, dacă condițiile nu sunt respectate user-ul este redirecționat la partea de enter session name și max number of buddies. Nu cred că e nevoie de explicații în plus.

Use case diagram



Asta e și mai simplă, trebuie doar arătat ce poate face user-ul, noi aici aveam mai multe tipuri de useri care moșteneau de la User (reprezentat prin săgeată) și tot ce puteau face ei în aplicație (pe partea de evenimente lucram pe atunci).

Acel extend se pune atunci când acea acțiune e posibilă, dar doar opțională, adică de exemplu la change status to going user-ul poate cumpăra bilet dacă evenimentul nu e free, sau la create avem opțiunea de a cumpăra bilet la evenimentele care nu sunt free, sau se pot modifica detaliile doar dacă vrea organizatorul, le poate lăsa și așa.

Mai există pe lângă <<extend>> și <<include>> care tot așa se leagă de o anumită acțiune, doar că cu săgeata invers, în celalaltă direcție, care după cum e și de la nume, include o altă acțiune care trebuie făcută după cea care o include, nu e opțională. (inițial am pus și din astea dar m-a pus să modific pentru că a considerat că nu trebuia așa făcut, dar de exemplu dacă am avea add ceva care necesită o validare am putea pune cu include validate acel ceva pentru că poate trebuie să îndeplinească anumite condiții pentru a fi folosit).