# ISP за UserService

## loadUserByUsername(String username)

За interface based пристапот се дефинираат карактеристиките:

C1: username е празен стринг

За functionality based пристапот се дефинираат карактеристиките:

C2: Се фрла UsernameNotFoundException

Секоја од карактеристиките може да се подели во два блока, т.е. еден блок Т (сите вредности за кои конкретниот исказ е точен) и еден блок F (сите вредности за кои конкретниот исказ е неточен).

а) Сите партиционирања го задоволуваат својството дисјунктност. За секое партиционирање објаснувањето е:

C1: Јасно е дека било кој string како вредност за параметарот username истовремено не може да е празен string и да не е празен string, односно тој string или е или не е празен.

C2: Јасно е дека е функцијата истовремено не може да го врати корисник кој има исто корисничко име како испратеното и да фрли го фрли исклучокот UsernameNotFoundException, односно или ќе фрли UsernameNotFoundException или пак ќе го врати корисникот.

б) Сите партиционирања го задоволуваат својството комплетност. За секое партиционирање објаснувањето е:

C1: Јасно е дека за било кој string како вредност за параметарот username важи дека или е или не е празен, а со партиционирањето се опфатени и двата случаи.

C2: Јасно е дека за функцијата важи дека или ќе фрли UsernameNotFoundException или пак ќе го врати корисникот, а со партиционирањето се опфатени и двата случаи.

в) Како основен тест се избира наједноставниот и најверојатниот случај, т.е. да не се праќа празен стринг како вредност за корисничкото име и да не се фрли исклучокот.

Тогаш основниот тест би бил: F F.

Потоа поради тоа што се користи Base Choice Coverage (BCC) критериумот, од основниот тест треба да произлезат уште 2 тестови.

Се добиваат следните тестови:

T F 🡪 Овој случај значи да username да е празен стринг и да не се фрли исклучок. Овој тест е можен само доколку полето user може да биде празен стринг што не е логично за внес во базата, па ќе го земеме овој случај како невозможен. За да е можен би било ТТ.

F T 🡪 Овој случај значи дека username има некоја вредност но се фрла исклучок. Овој тест е остварлив бидејќи доколку не постои корисник со таков username тогаш се фрла исклучокот.

Тестни случаи:

* Влез: username=”admin” Очекуван излез: се враќа корисникот со корисничко име “admin”
* Влез: username=”” Очекуван излез: се фрла UsernameNotFoundException
* Влез: username=”bojan08” Очекуван излез: се фрла UsernameNotFoundException

## Register (UserDTO userDTO)

За interface based пристапот се дефинираат карактеристиките:

C1: userDTO.username e null

C2: userDTO.password e null

За functionality based пристапот се дефинираат карактеристиките:

C3: userDTO.password== userDTO.repeatedPassword

C4: се фрла UserAlreadyExistsException

Секоја од карактеристиките може да се подели во два блока, т.е. еден блок Т (сите вредности за кои конкретниот исказ е точен) и еден блок F (сите вредности за кои конкретниот исказ е неточен).

а) Сите партиционирања го задоволуваат својството дисјунктност. За секое партиционирање објаснувањето е:

C1: Јасно е дека било кој string како вредност за параметарот userDTO.username истовремено не може да е null и да не е null, односно тој string или е или не е null.

C2: Јасно е дека било кој string како вредност за параметарот userDTO.password истовремено не може да е null и да не е null, односно тој string или е или не е null.

C3: Јасно е дека за било кој објект како вредност за параметарот userDTO не може истовремено да важи userDTO.password== userDTO.repeatedPassword и userDTO.password!= userDTO.repeatedPassword, односно или важи userDTO.password== userDTO.repeatedPassword или не

C4: Јасно е дека за било кој објект како вредност за параметарот userDTO не може истовремено да се регистрира корисникот и да се фрли исклучокот UserAlreadyExistsException, односно или се фрла исклучокот или не.

б) Сите партиционирања го задоволуваат својството комплетност. За секое партиционирање објаснувањето е:

C1: Јасно е дека за било кој string како вредност за параметарот userDTO.username важи дека или е или не е null, а со партиционирањето се опфатени и двата случаи.

C2: Јасно е дека за било кој string како вредност за параметарот userDTO.password важи дека или е или не е null, а со партиционирањето се опфатени и двата случаи.

C3: Јасно е дека за било кој објект како вредност за параметарот userDTO важи или userDTO.password== userDTO.repeatedPassword или userDTO.password!= userDTO.repeatedPassword, а со партиционирањето се опфатени и двата случаи.

С4: Јасно е дека за функцијата важи дека или ќе фрли UsernameNotFoundException или пак ќе го регистрира корисникот, а со партиционирањето се опфатени и двата случаи.

в) Како основен тест се избира наједноставниот и најверојатниот случај, т.е. username и password да се различни од null, лозниките да се совпаѓаат и да не се фрли исклучок

Тогаш основниот тест би бил: F F T F.

Потоа поради тоа што се користи Base Choice Coverage (BCC) критериумот, од основниот тест треба да произлезат уште 4 тестови.

T F T F 🡪 Се менува само првата вредност. Тестот е Feasible.

F T T F 🡪 Се менува само втората вредност. Тестот е Feasible.

F F F F 🡪 Се менува само третата вредност. Тестот е Feasible.

T F T T 🡪 Се менува само четвртата вредност. Тестот е Feasible.

Тестни случаи:

* Влез: UserDTO(name=”Darko”, surname=”Sasanski”, username=”dare431”, password=”Password@123”, repeatPassword=”Password@123”, role= ROLE\_USER), Очекуван излез: се враќа новиот регистриран корисник
* Влез: UserDTO(name=”Darko”, surname=”Sasanski”, username=null, password=”Password@123”, repeatPassword=”Password@123”, role= ROLE\_USER), Очекуван излез: се фрла InvalidArgumentsException
* Влез: UserDTO(name=”Darko”, surname=”Sasanski”, username=”dare431”, password=null, repeatPassword=”Password@123”, role= ROLE\_USER), Очекуван излез: се фрла InvalidArgumentsException
* Влез: UserDTO(name=”Darko”, surname=”Sasanski”, username=”dare431”, password=”Password@123”, repeatPassword=”Password@12”, role= ROLE\_USER), Очекуван излез: се фрла PasswordsDoNotMatchException
* Влез: UserDTO(name=”Darko”, surname=”Sasanski”, username=”admin”, password=”Password@123”, repeatPassword=”Password@123”, role= ROLE\_USER), Очекуван излез: се фрла UsernameAlreadyExistsException

1. authorizePendingAdmin(String username)

За interface based пристапот се дефинираат карактеристиките:

C1: username е празен стринг

За functionality based пристапот се дефинираат карактеристиките:

C2: Се фрла UsernameNotFoundException

C3: user-от нема улога ROLE\_PENDING\_ADMIN

Секоја од карактеристиките може да се подели во два блока, т.е. еден блок Т (сите вредности за кои конкретниот исказ е точен) и еден блок F (сите вредности за кои конкретниот исказ е неточен).

а) Сите партиционирања го задоволуваат својството дисјунктност. За секое партиционирање објаснувањето е:

C1: Јасно е дека било кој string како вредност за параметарот username истовремено не може да е празен string и да не е празен string, односно тој string или е или не е празен.

C2: Јасно е дека е функцијата истовремено не може да го врати корисник кој има исто корисничко име како испратеното и да фрли го фрли исклучокот UsernameNotFoundException, односно или ќе фрли UsernameNotFoundException или пак ќе го врати корисникот.

C3: Јасно е дека корисникот со корисничкото име пратено како параметар на функцијата не може истовремено да ја има улогата ROLE\_PENDING\_ADMIN и да ја нема, односно корисникот или има улога ROLE\_PENDING\_ADMIN или има некоја од останатите улоги.

б) Сите партиционирања го задоволуваат својството комплетност. За секое партиционирање објаснувањето е:

C1: Јасно е дека за било кој string како вредност за параметарот username важи дека или е или не е празен, а со партиционирањето се опфатени и двата случаи.

C2: Јасно е дека за функцијата важи дека или ќе фрли UsernameNotFoundException или пак ќе го врати корисникот, а со партиционирањето се опфатени и двата случаи.

С3: Јасно е дека корисникот со корисничкото име пратено како параметар на функцијата или има улога ROLE\_PENDING\_ADMIN или има некоја од останатите улоги, а со партиционирањето се опфатени и двата случаи.

в) Како основен тест се избира наједноставниот и најверојатниот случај, т.е. username да не е празен string, да не се фрли исклучокот UsernameNotFoundException и корисникот да има улога ROLE\_PENDING\_ADMIN.

Тогаш основниот тест би бил: F F F.

Потоа поради тоа што се користи Base Choice Coverage (BCC) критериумот, од основниот тест треба да произлезат уште 3 тестови.

T F F 🡪 Овој случај значи да username да е празен стринг и да не се фрли исклучок. Овој тест е можен само доколку полето user може да биде празен стринг што не е логично за внес во базата, па ќе го земеме овој случај како невозможен. За да е можен потребно е да ги промениме и другите две вредности и би добиле Т Т Т.

F T F 🡪 Се менува само втората вредност. Овој тест е Feasible.

F F T 🡪 Се менува само третата вредност. Овој тест е Feasible.

Тестни случаи:

* Влез: username=”pendingAdmin” Очекуван излез: се враќа корисникот со корисничко име “pendingAdmin” со нова улога
* Влез: username=”” Очекуван излез: се фрла UsernameNotFoundException
* Влез: username=”bojan08” Очекуван излез: се фрла UsernameNotFoundException
* Влез: username=”dare431” Очекуван излез: се фрла InvalidArgumentsException

## addToFavourites(String username, Long locationId)

За interface based пристапот се дефинираат карактеристиките:

C1: username е празен стринг

C2: locationId е број поголем или еднаков од 0

За functionality based пристапот се дефинираат карактеристиките:

C3: корисникот веќе ја има додадено локацијата во листата

Секоја од карактеристиките може да се подели во два блока, т.е. еден блок Т (сите вредности за кои конкретниот исказ е точен) и еден блок F (сите вредности за кои конкретниот исказ е неточен).

а) Сите партиционирања го задоволуваат својството дисјунктност. За секое партиционирање објаснувањето е:

C1: Јасно е дека било кој string како вредност за параметарот username истовремено не може да е празен string и да не е празен string, односно тој string или е или не е празен.

C2: Јасно е дека било кој број како вредност за параметарот locationId истовремено не може да е поголем или еднаков од 0 и да не е, односно тој број или е или не е поголем или еднаков од 0.

C3: Јасно е дека корисникот со корисничкото име пратено како параметар на функцијата не може истовремено да ја има веќе додадена локацијата со id испратено како параметар на функцијата и да ја нема, односно корисникот или ја има или ја нема додадено веќе локацијата во листата.

б) Сите партиционирања го задоволуваат својството комплетност. За секое партиционирање објаснувањето е:

C1: Јасно е дека за било кој string како вредност за параметарот username важи дека или е или не е празен, а со партиционирањето се опфатени и двата случаи.

C2: Јасно е дека за било кој број како вредност за параметарот locationId важи дека или е или не е поголем или еднаков од 0, а со партиционирањето се опфатени и двата случаи.

С3: Јасно е дека корисникот со корисничкото име пратено како параметар на функцијата или ja има додадена или не веќе додадена локацијата со id испратено како параметар на функцијата, а со партиционирањето се опфатени и двата случаи.

в) Како основен тест се избира наједноставниот и најверојатниот случај, т.е. username да не е празен string, locationId да е број поголем или еднаков од 0 и корисникот да ја нема претходно додадена локацијата.

Тогаш основниот тест би бил: F Т F.

Потоа поради тоа што се користи Base Choice Coverage (BCC) критериумот, од основниот тест треба да произлезат уште 3 тестови.

T T F 🡪 Се менува само првата вредност. Овој тест е Feasible.

F F F 🡪 Се менува само втората вредност. Овој тест е Feasible.

F F Т 🡪 Се менува само третата вредност. Овој тест е Feasible.

Тестни случаи:

* Влез: username=”andrejT”, locationId=1 Очекуван излез: се враќа корисникот со корисничко име “andrejT” со aжурирана листа на омилени локации
* Влез: username=”””, locationId=1 Очекуван излез: се фрла UsernameNotFoundException
* Влез: username=”dare431”, locationId=-1 Очекуван излез: се фрла LocationNotFoundException
* Влез: username=”admin”, locationId=1 Очекуван излез: се враќа корисникот со корисничко име “admin” но со непроменета листа на омилени локации

## addToVisited(String username, Long locationId)

За interface based пристапот се дефинираат карактеристиките:

C1: username е празен стринг

C2: locationId е број поголем или еднаков од 0

За functionality based пристапот се дефинираат карактеристиките:

C3: корисникот веќе ја има додадено локацијата во листата

Секоја од карактеристиките може да се подели во два блока, т.е. еден блок Т (сите вредности за кои конкретниот исказ е точен) и еден блок F (сите вредности за кои конкретниот исказ е неточен).

а) Сите партиционирања го задоволуваат својството дисјунктност. За секое партиционирање објаснувањето е:

C1: Јасно е дека било кој string како вредност за параметарот username истовремено не може да е празен string и да не е празен string, односно тој string или е или не е празен.

C2: Јасно е дека било кој број како вредност за параметарот locationId истовремено не може да е поголем или еднаков од 0 и да не е, односно тој број или е или не е поголем или еднаков од 0.

C3: Јасно е дека корисникот со корисничкото име пратено како параметар на функцијата не може истовремено да ја има веќе додадена локацијата со id испратено како параметар на функцијата и да ја нема, односно корисникот или ја има или ја нема додадено веќе локацијата во листата.

б) Сите партиционирања го задоволуваат својството комплетност. За секое партиционирање објаснувањето е:

C1: Јасно е дека за било кој string како вредност за параметарот username важи дека или е или не е празен, а со партиционирањето се опфатени и двата случаи.

C2: Јасно е дека за било кој број како вредност за параметарот locationId важи дека или е или не е поголем или еднаков од 0, а со партиционирањето се опфатени и двата случаи.

С3: Јасно е дека корисникот со корисничкото име пратено како параметар на функцијата или ja има додадена или не веќе додадена локацијата со id испратено како параметар на функцијата, а со партиционирањето се опфатени и двата случаи.

в) Како основен тест се избира наједноставниот и најверојатниот случај, т.е. username да не е празен string, locationId да е број поголем или еднаков од 0 и корисникот да ја нема претходно додадена локацијата.

Тогаш основниот тест би бил: F Т F.

Потоа поради тоа што се користи Base Choice Coverage (BCC) критериумот, од основниот тест треба да произлезат уште 3 тестови.

T T F 🡪 Се менува само првата вредност. Овој тест е Feasible.

F F F 🡪 Се менува само втората вредност. Овој тест е Feasible.

F F Т 🡪 Се менува само третата вредност. Овој тест е Feasible.

Тестни случаи:

* Влез: username=”aleksej”, locationId=1 Очекуван излез: се враќа корисникот со корисничко име “aleksej” со aжурирана листа на посетени локации
* Влез: username=”””, locationId=1 Очекуван излез: се фрла UsernameNotFoundException
* Влез: username=”dare431”, locationId=-1 Очекуван излез: се фрла LocationNotFoundException
* Влез: username=”admin”, locationId=1 Очекуван излез: се враќа корисникот со корисничко име “admin” но со непроменета листа на посетени локации