Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра ЕОМ

Лабораторна робота №1

з дисципліни: «Інженерія програмного забезпечення»

на тему:” Система бронювання та купівля залізничних квитків.”

Виконав: ст. гр. КІ-32
Гунько Б. О.
Прийняв:
Цигилик Л.О.

Львів – 2020

Тема: UML діаграми варіантів використання та UML діаграми активності.

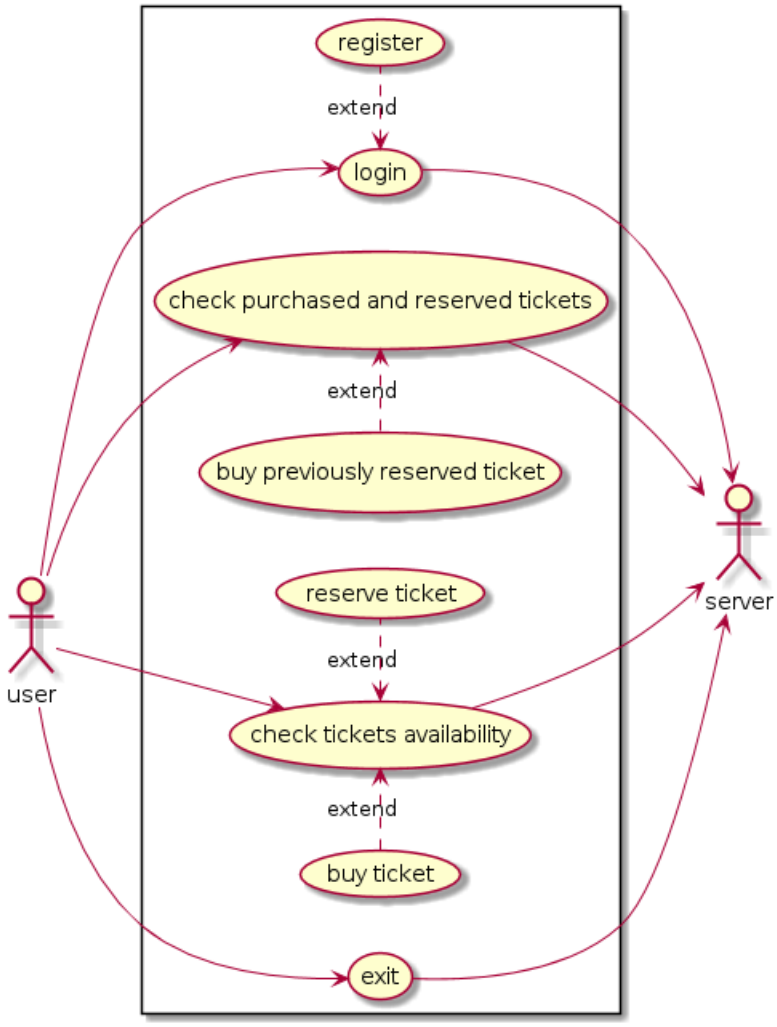
Мета: Освоїти принципи створення UML діаграм варіантів використання що описують сценарій роботи системи (Use case diagram) та діаграм активності для клієнтської та серверної частин.

Постаноска задачі: Розробити UML діаграми сценаріїв роботи клієнтської та серверної частин а також розробити діаграми активності для кожного сценарію. Кількість UML діаграм повинна бути не менше – 8.

Моя система даватиме користувачам змогу переглядати наявність квитків на потрібний їм потяг, показуватиме ціни на ці квитки а також дозволить купити квитки або ж забронювати їх з подальшою можливістю їх покупки.У цій програмі передбачений процес автоизації та процес реєстрації.Усі дані про розклад потягів, доступність квитків та їх цін зберігатимуться на сервері, клієнська частина буде лише відображати отримані від сервера дані.Сервер ж у свою чергу буде виконувати всю логіку роботи системи (пошук потягів, квитків, бронювання, купівля…).

Код до кожної з діаграм наведено в додатку, який знаходиться в кінці звіту.

UML діаграма сценарію роботи клієнтської частини



Опис діаграми клієнтської частини

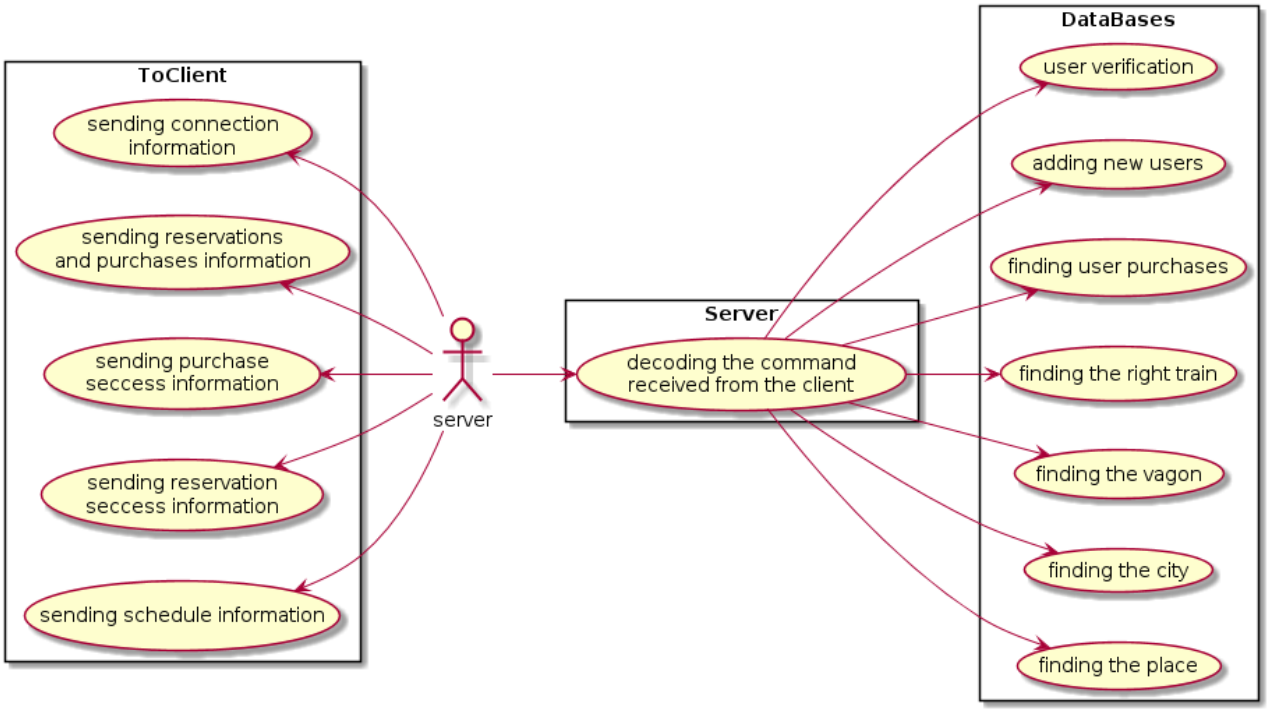
Дана діаграма зображає можливі сценарії роботи клієнтської частини. Оскільки система взамодіє з користувачем і сервером, на ній присутні два актори: Client і Server. Операція воду в систему(Log in) передбачає можливість реєстрація нового користувача(Register), на діаграмі вона показана як extend оскільки є не обовязковою.Також передбачені такі сценарії використання:

- Перегляд раніше куплених квитків а також перегляд попередньо забронюваних квитків (check purchased and reserved tickets) з можливістю їх покупки(buy previously reserved tickets) . Сценарій покупки попередньо зарезервованих квитків також є розширенням (extend) адже незавжди зарезервовані квитки будуть куплені.
- Перегляд наявності квитків на потрібний користувачу потяг (check tickets availability) з можливістю купити(buy ticket) або ж зарезервувати (reserve

ticket) потрібні квитки. Купівлята резервація є розширеннями цього сценарію адже не завжди користувач знайде потрібні йомуквитки.

- Вихід (делогінізація) з системи.

UML діаграма сценарію роботи серверної частини

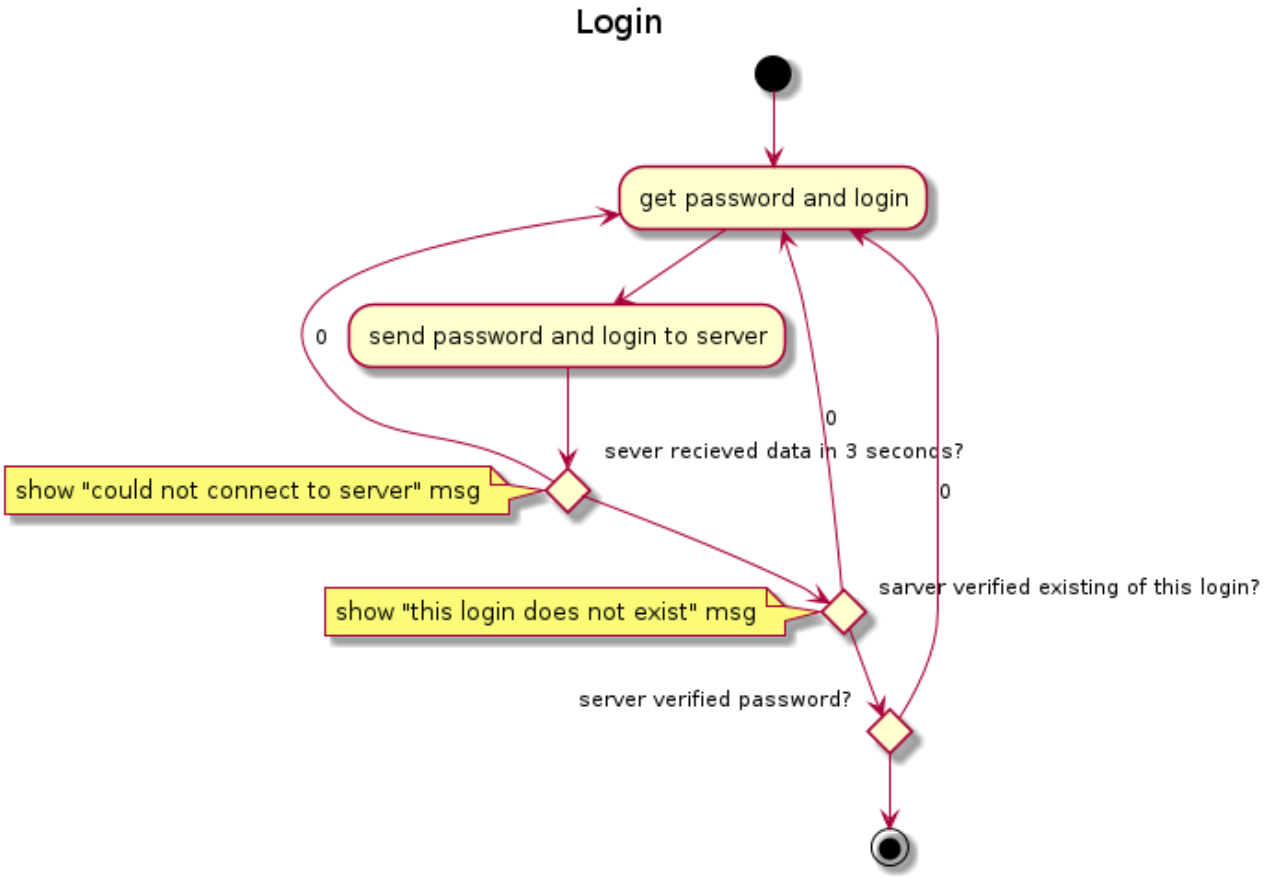


Опис діаграми серверної частини

Дана діаграма зображає можливі сценарії роботи серверної частини. Сервер може отримувати дані від клієнта а також передавати дані йому.Сервер декодує отриману від клієнта інформацію щоб знати що саме йому потрібно робити (в якій саме базі даних шукати, які саме дані шукати).При відправці даних від сервера до клієнта сервер приводить дані до відомого наперед формату. Це зроблено для того, щоб клієнт міг їх правильно інтерпретувати та відобразити.Також сервер взаємодіє з базами даних, виконує операції зчитування та запису інформації.Сервер виконує пошук потрібної інформації в базах даних для подальшого надсилання цієї інформації до клієнта.Обмін інформацією відбувається за допомогою протоколу TCP IP, що дає змогу швидко та зручно обмінюватись даними.

Діаграми активності клієнської частини та їх опис

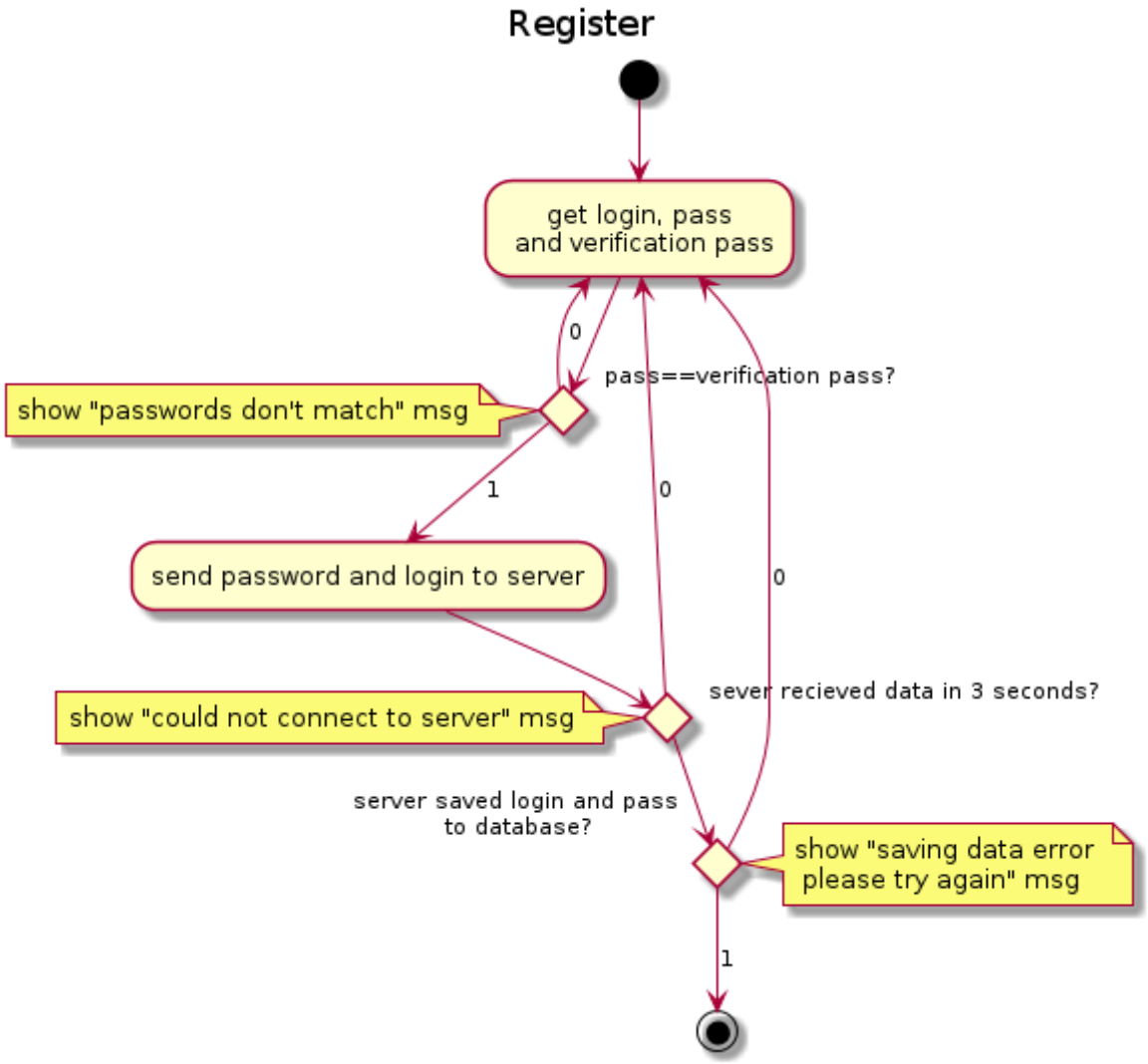
Процес входу в систему



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при спробі користувача увійти в систему.В процесі авторизації передбачені перевірка на отримання сервером даних, перевірка на існування логіну та коректність паролю. Як можемо бачити з діаграми клієнська частина не робить цих перевірок а лише приймає рішення про перехід (або не перехід) до наступного кроку базуючись на отриманій від сервера інформації.При невірності логіну або паролю користувач матиме змогу спробувати авторизуватись ще раз.

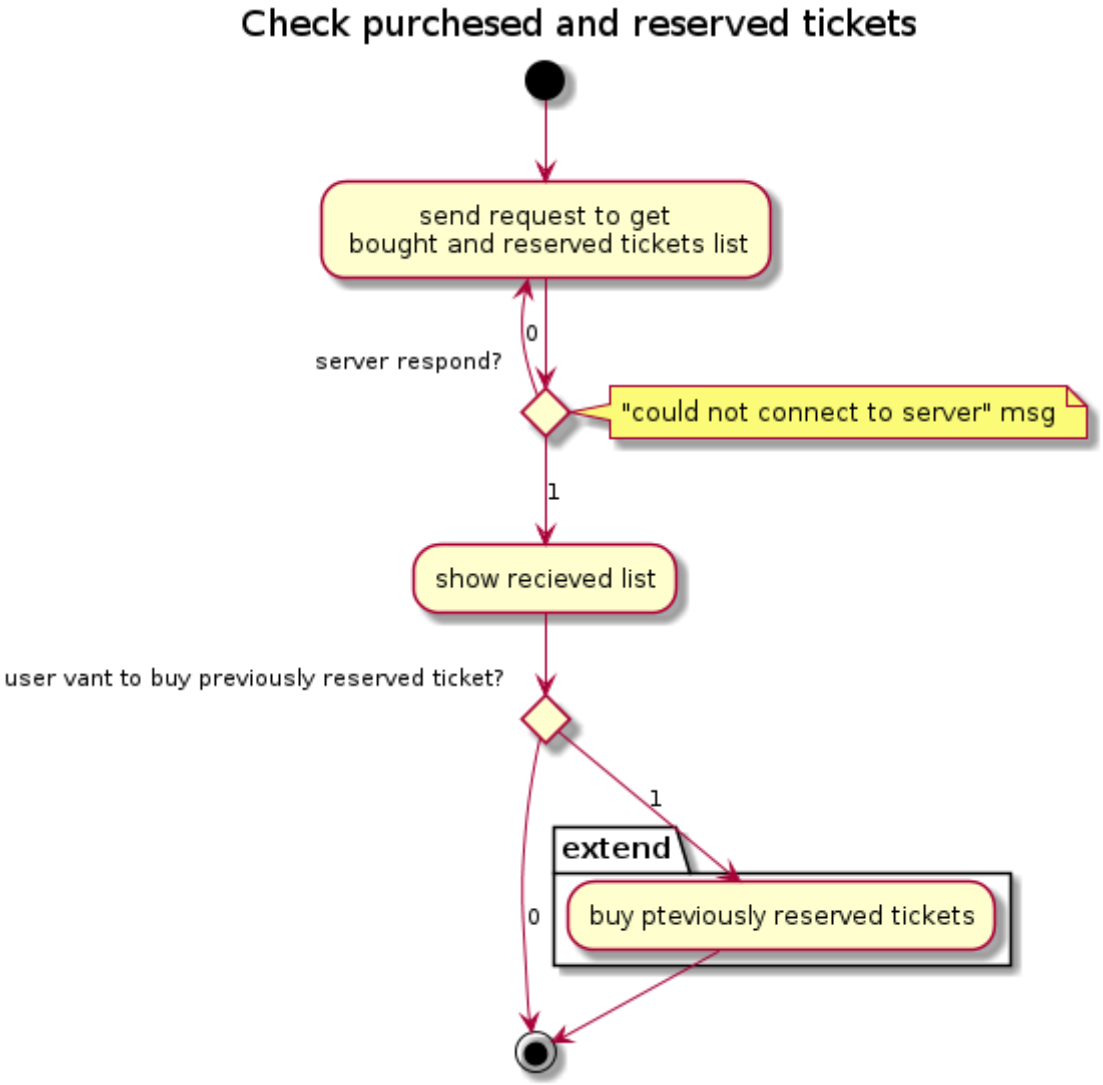
Процес реєстрації



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при спробі реєстрації нового користувача.В процесі реєстрації передбачені перевірка на отримання сервером даних та перевірка на співпадіння паролів які увів користувач.З діаграми видно що перевірка на співпадіння паролей виконується на стороні клієнтської частини, зроблено це з метою оптимізації швидкодії програми, адже немає сенсу передавати два паролі на сервер для їх порівняння бо це збільшує обєми інформації яку потрібно передавати.При різності паролів буде виведено повідомлення про неспівпадіння введених користувачем паролів, а користувач отримає змогу спробувати зареєструватись повторно.

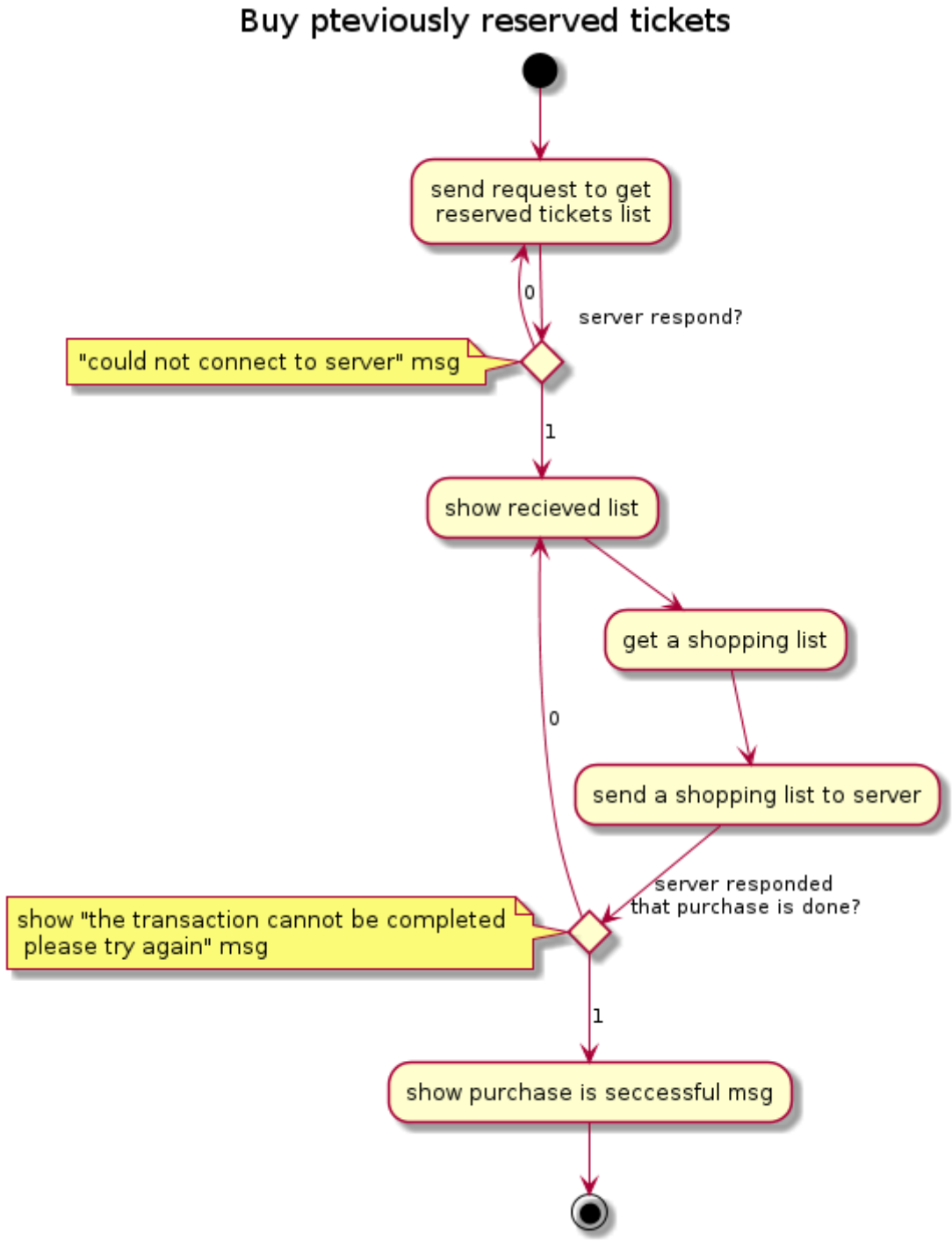
Процес перегляду куплених та попередньо заброньованих квитків



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при перегляді куплених та попередньо заброньованих квитків .Під час виконання цього процесу клієнт надсилає запит на отримання спискуквитків які користувач купив або забронював. При отриманні цього списку клієнт відобразить їх на екран. Крім змоги переглянути ці квитки користувач матиме змогу натиснути кнопку “купити заброньований квиток” після чого він побачить список заброньованих ним квитків і матиме змогу придбати один з них.Дії які потрібно виконати під час процесу покупки попередньо заброньованого квитка будуть описані у наступній діаграмі.При небажанні купляти один з заброньованих квитків користувач зможе повернутись у головне меню.

Процес покупки попередньо заброньованих квитків

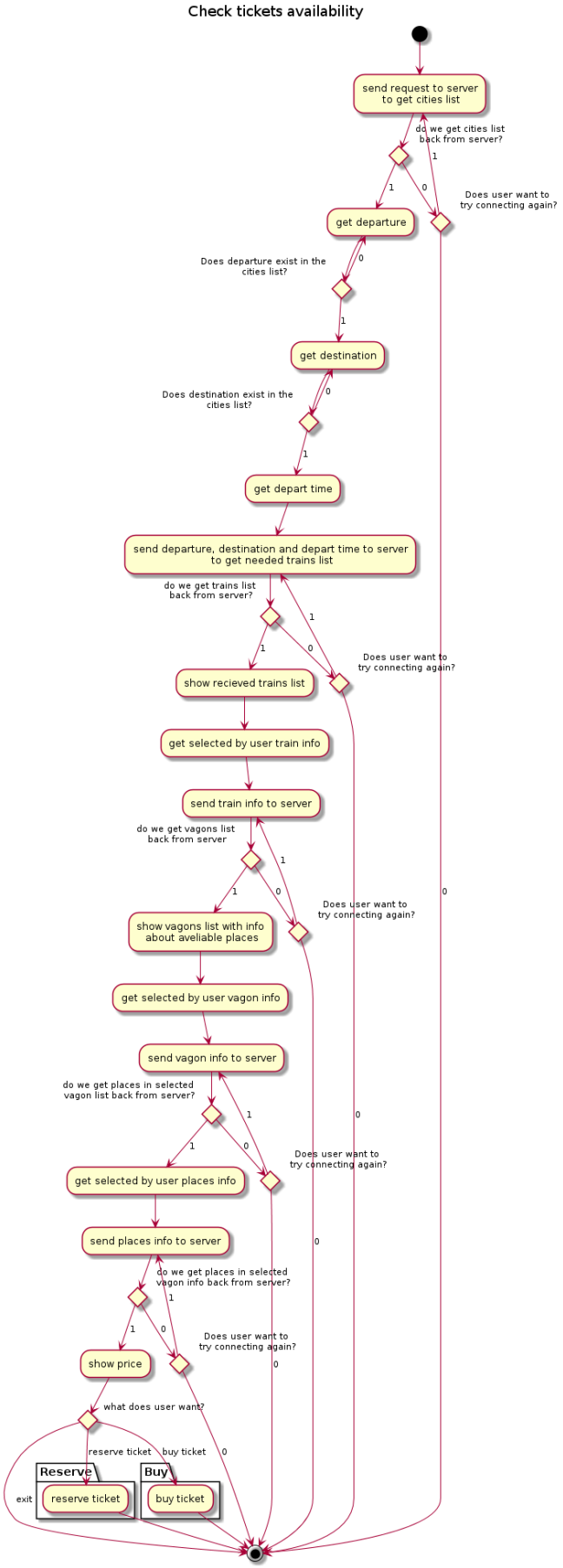


Опис діаграми

Ця діаграма показує послідосність дій які виконуються при покупці попередньо заброньованих квитків .Після того як користувач вибрав квитки які він бажає купити буде сформовано їх список, після чого цей список буде відправлено на сервер.Якщо клієнт не зміг зєднатись з сервером на екрані користувача зявиться відповідне повідомлення, після чого користувач зможе спробувати ще раз.Якщо ж дані надійшли на сервер та інформація про покупку квитків була успішно опрацьована та занесена в базу даних користувач отримає

повідомлення про успішність покупки, після чого він зможе перейти у головне меню.

Процес перегляду доступних на потрібний потяг квитків

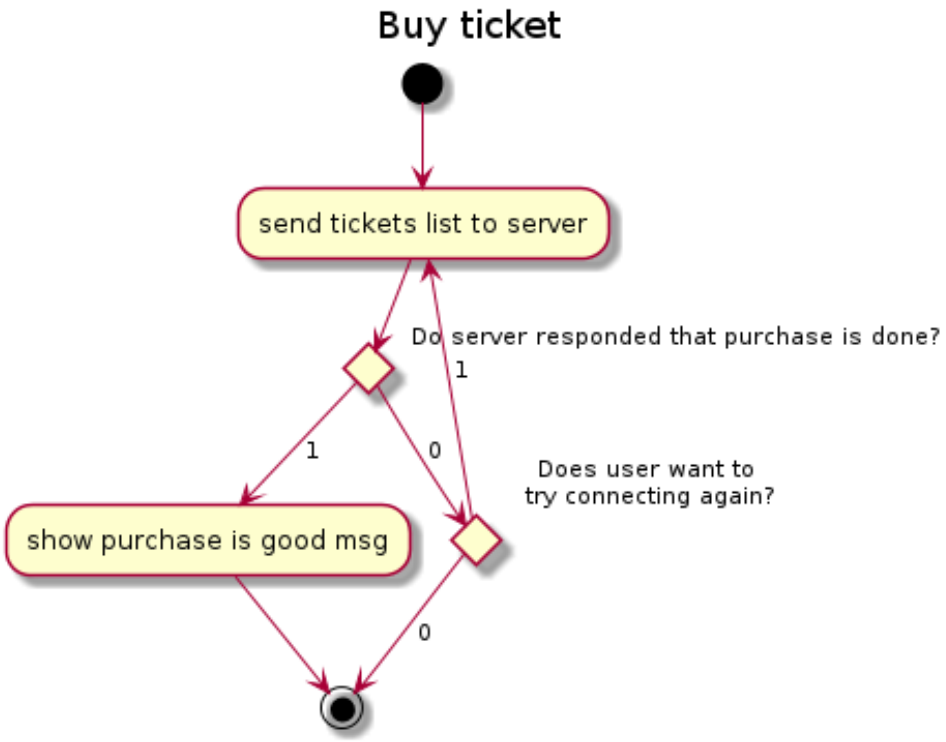


Опис діаграми

Ця діаграма показує послідосність дій які виконуються при перегляді доступних на потрібний потяг квитків.Як можемо бачити на діаграмі цей процес включає в себе обирання клієнтом точки відправлення, точки прибуття, дати відправлення.

Після цього отримані дані відправляються на сервер для отримання списку поїздів. Після чого список поїздів відображається на екрані користувача.Користувач обирає потрібний йому потяг, вагон та місце після чого він може вийти в головне меню або ж купити чи зарезервувати квитки.Діаграми процедур покупки та резервацій квитків разом з їх описом наведені нище.

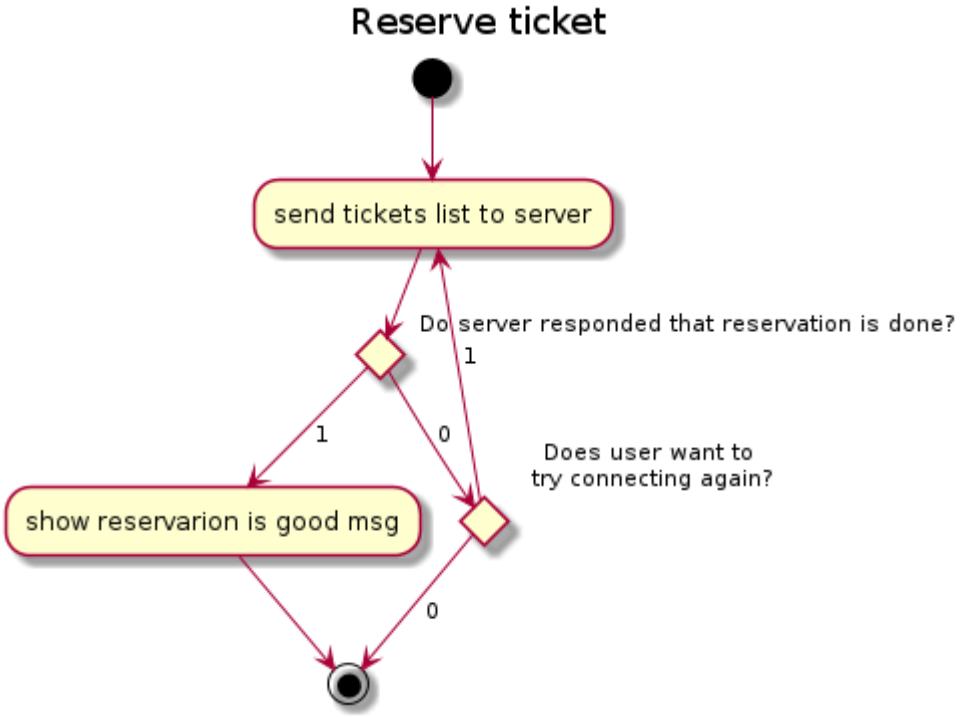
Процес покупки доступних на потрібний потяг квитків



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при покупці доступних на потрібний потяг квитків. Клієнська частина надсилає список обраних клієнтом квитків на сервер після чого очікує відповіді від сервера.Якщо сервер отримав дані та повідомив про успішну покупку, то клієнтська частина відобразить повідомлення про успішність покупки. Якщо ж сервер не зміг отримати дані або ж не зміг здійснити покупку користувач отримає відповідне повідомлення та матиме змогу здійснити покупку повторно.

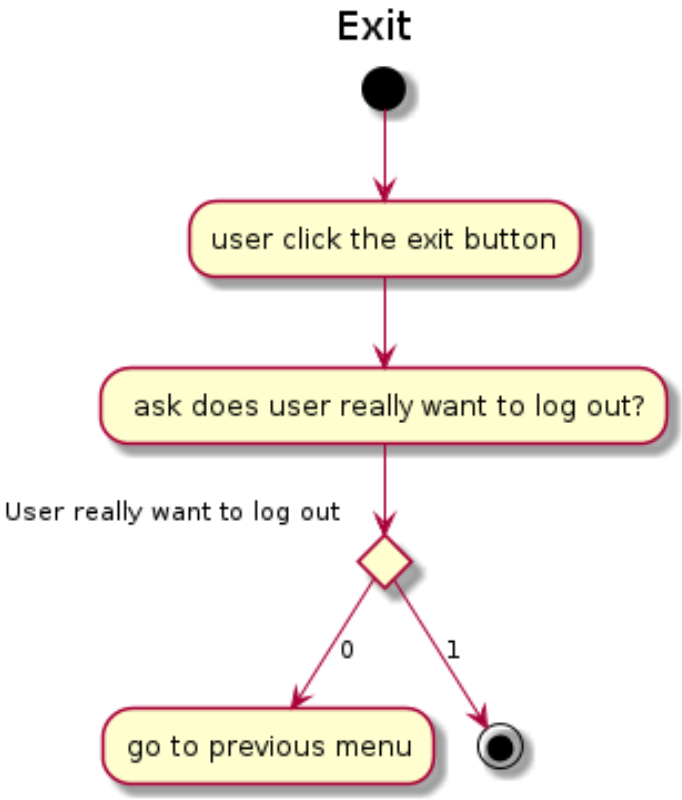
Процес резервації доступних на потрібний потяг квитків



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при резервації доступних на потрібний потяг квитків. Клієнська частина надсилає список обраних клієнтом квитків на сервер після чого очікує відповіді від сервера.Якщо сервер отримав дані та повідомив про успішну резервацію, то клієнтська частина відобразить повідомлення про успішність резервації. Якщо ж сервер не зміг отримати дані або ж не зміг здійснити резервацію користувач отримає відповідне повідомлення та матиме змогу здійснити резервацію повторно.

Процес виходу з системи

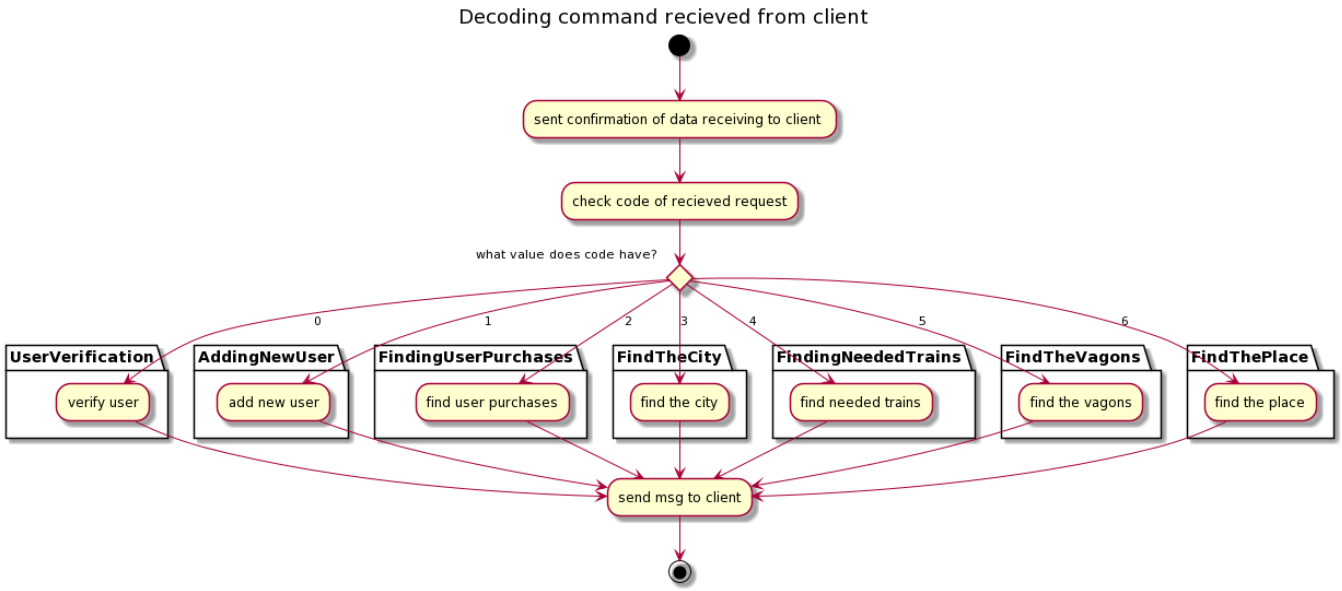


Опис діаграми

Ця діаграма показує послідосність дій які виконуються при виході з системи.Коли користувач натискає кнопку “Вихід” він повинен буде підтвердити своє бажання вийти з системи шляхом натискання книпки “Так”. У будь якому іншому випадку вихід з системи виконано не буде. Зроблено це для запобігання помилковим натисканням.

Діаграми активності серверної частини та їх опис

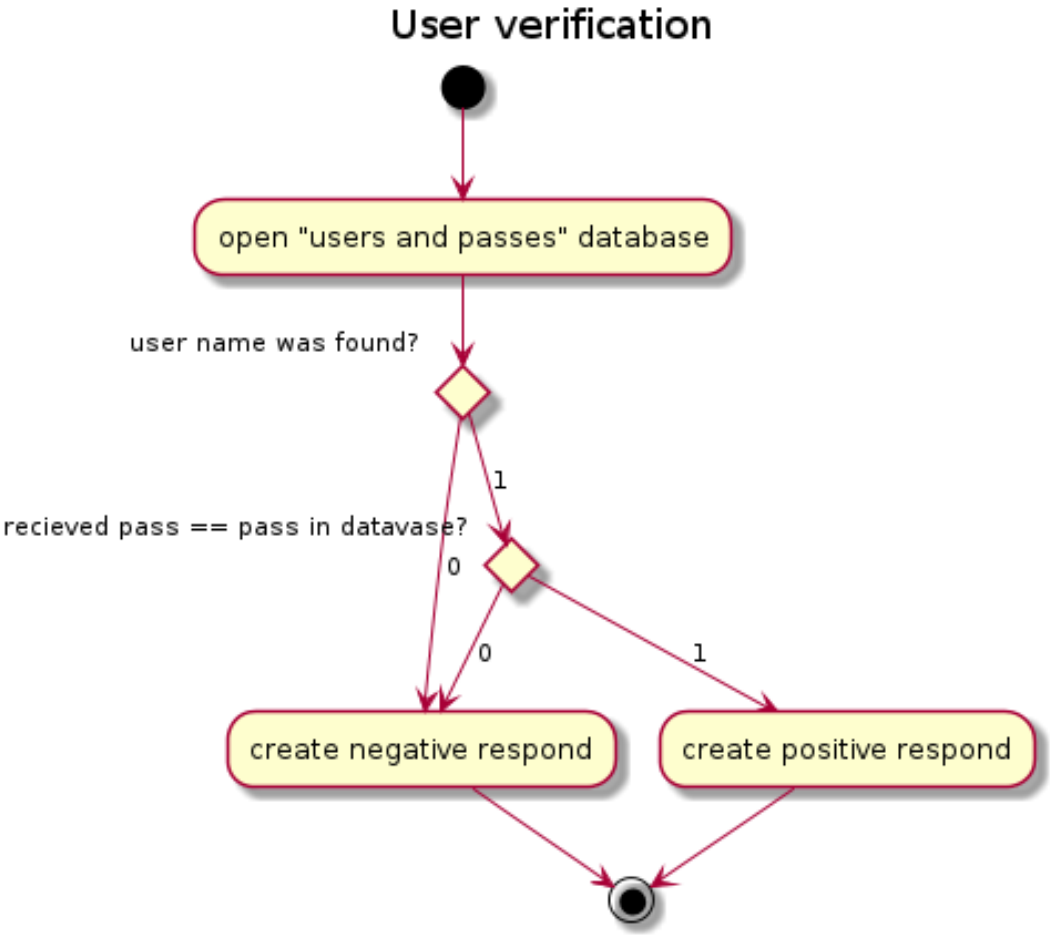
Процес декодування отриманої від клієнта команди



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при декодіровці отриманих від клієнта повідомлень.У кожної команди є свій унікальний код що допомагає однозначно визначити які дії потрібно виконати для отримання бажаного результату.

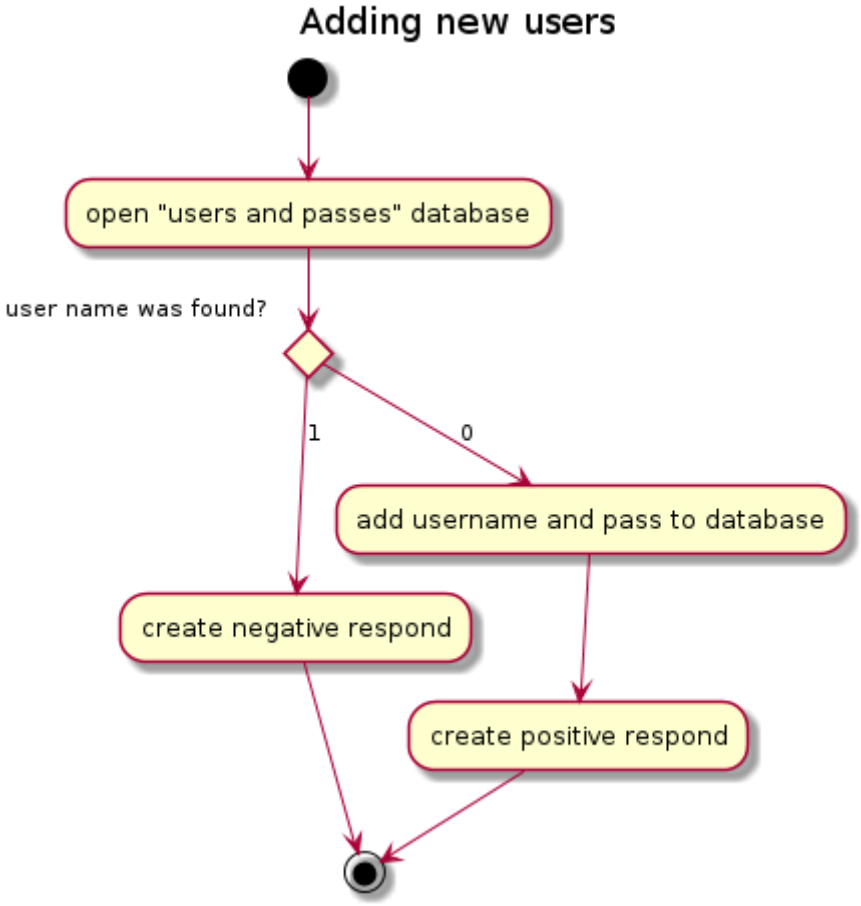
Процес верифікації користувача



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при спробі користувача увійти в систему.В процесі авторизації сервер отримує логін та пароль від клієнської частини після чого він відкриває базу даних де зберігаються логіни та паролі та перевіряє наявність отриманого від користувача логіну у базі даних. При відсутності логіну у базі даних сервер відправить відповідне повідомлення до клієнта. Якщо ж логін існує в базі даних відбувається перевірка на коректність паролю, якщо пароль невірний сервер повідомить про це клієнта, якщо ж верифікація пройшла успішн осервер надісле повідомлення про це до користувача.

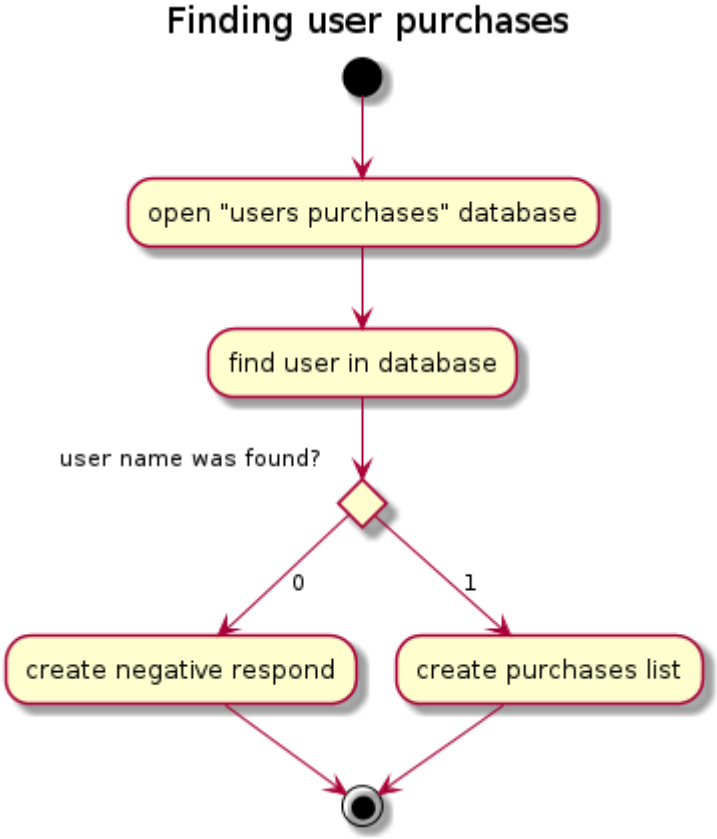
Процес додавання нового користувача



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при додаванні нових користувачів.Після того як сервер отримує логін та пароль від клієнта він відкриває базу даних де зберігаються логіни та паролі та перевіряє наявність отриманого логіну у цій базі даних. Якщо логін був знайдени, то сервер надішле клієнту повідомлення що такий користувач уже існує.Якщо ж логін не знайдено в базі даних то він разом з паролем буде доданий до цієї бази даних.

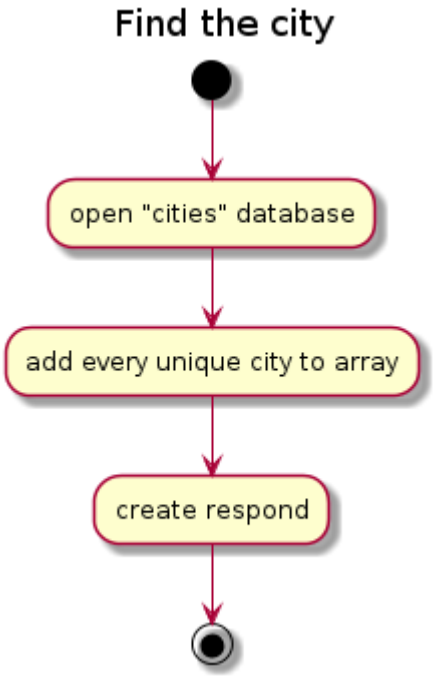
Процес пошуку попередньо заброньованих та куплених квиктів



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при пошуку покупок та резервацій користувача.Під час виконання цього процесу червер відкриває базу даних з покупками користувачів і шукає у ній потрібного користувача.Якщо користувач не був знайдений то до клієнта буде відправлено пустий список, якщо ж цей користувач здійснював покупки чи резервації раніше він буде знайдений у базі даних, після чого буде сформовано та надіслано до клієнта список його покупок та резервацій.

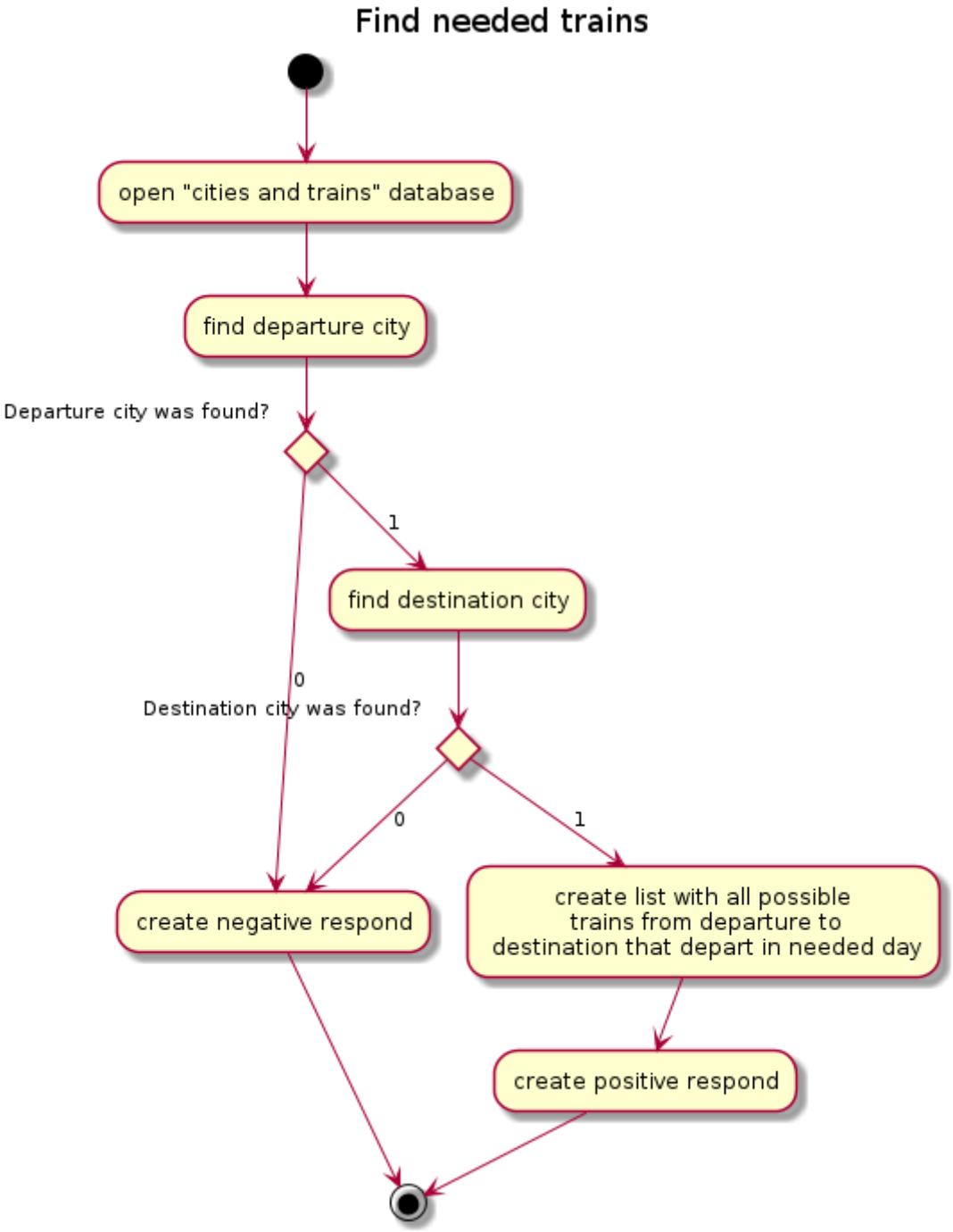
Процес пошуку міст



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при пошуку списку міст.Спочатку сервер відкриває базу даних з містами після чого формує список з усіх міст (без повторень) після чого надсилає цей список клієнтові.

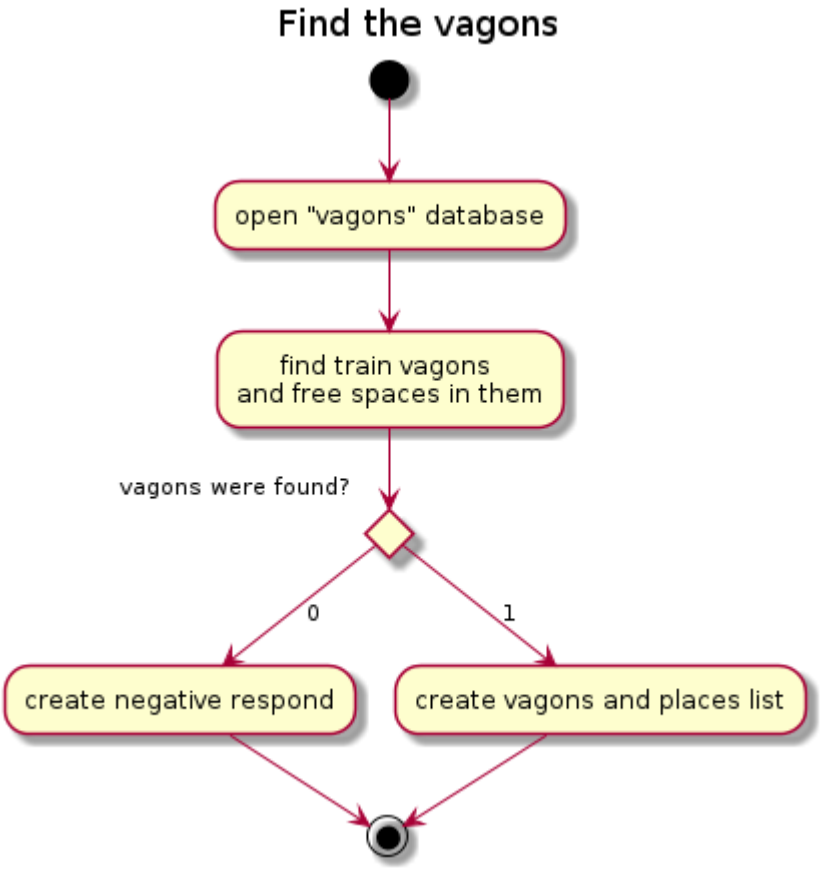
Процес пошуку потрібного потягу



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при пошуку потяга з потрібними користувачу параметрами.Спочатку сервер відкриває базу даних, в якій зберігаються дані про поїзди. Сервер знаходить потяг який проїзджає через точку відправлення користувача та прямує до пункту призначення користувача. Якщо такий потяг не буде знайдено то сервер повідомить клієнта про це. Якщо ж буде знайдено підходящі потяги(потяг) то сервир створить список з цих потягів та надішле його до клієнта.

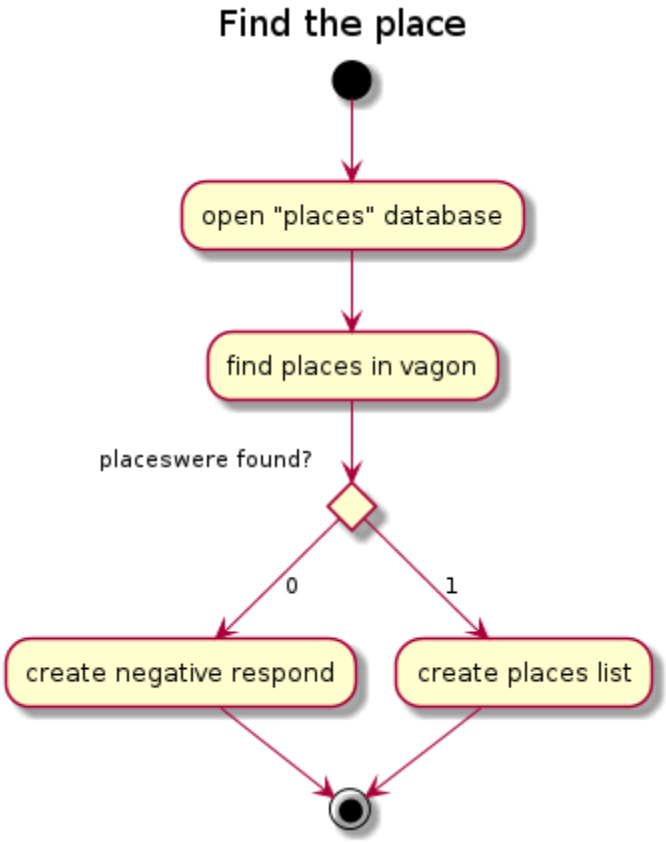
Процес пошуку вагону



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при пошуку вагону в потрібному нам потязі.Спочатку сервер відкриває базу даних з списком вагонів даного потягу, після чого формує список з вагонів та надсилає їх до клієнта.

Процес пошуку місця



Опис діаграми

Ця діаграма показує послідосність дій які виконуються при пошуку місця в потрібному вагоні.Спочатку сервер відкриває базу даних з списком місць у даному вагоні, після чого формує список з місць та надсилає їх до клієнта.

Додотак з кодами до діаграм

UML діаграма сценарію роботи клієнтської частини

```
@startuml
left to right direction
skinparam packageStyle rectangle
actor user
actor server
rectangle {
user --> (login)
(register).left.>(login) : extend
user-->(check purchased and reserved tickets)
(buy previously reserved ticket) .> (check purchased and reserved tickets) : extend
user --> (check tickets availability)
(buy ticket) .right.> (check tickets availability):extend
(reserve ticket) .left.> (check tickets availability):extend
user --> (exit)
(login) --> server
(check purchased and reserved tickets) --> server
(check tickets availability) --> server
(exit) --> server
}
@enduml
```

UML діаграма сценарію роботи серверної частини

```
@startuml
left to right direction
skinparam packageStyle rectangle
actor server
rectangle ToClient{
(sending connection \ninformation)<-down-server
(sending reservations \nand purchases information)<-down-server
(sending purchase \nseccess information) <-down-server
(sending reservation \nseccess information) <-down-server
(sending schedule information) <-down-server
}
rectangle Server{
server-down->(decoding the command \nreceived from the client)
}
rectangle DataBases{
(decoding the command \nreceived from the client)-down->(user verification)
(decoding the command \nreceived from the client)-down->(adding new users)
(decoding the command \nreceived from the client)-down->(finding user purchases)
(decoding the command \nreceived from the client)-down->(finding the right train)
(decoding the command \nreceived from the client)-down->(finding the vagon)
  (decoding the command \nreceived from the client)-down->(finding the city)
  (decoding the command \nreceived from the client)-down->(finding the place)
}
@enduml
```

Діаграми активності клієнської частини та їх опис

Процес входу в систему

```
@startuml
title Login
(*) -->"get password and login"
-->"send password and login to server"
if "sever recieved data in 3 seconds?" then
note left: show "could not connect to server" msg
  -left->[0]"get password and login"
else
  if "sarver verified existing of this login?" then
    note left: show "this login does not exist" msg
    -->[0]"get password and login"
  else
    if "server verified password?" then
      -->[0]"get password and login"
```

```
   else
     -->(*)
   endif
  endif
endif
@enduml
```

Процес реєстрації

```
@startuml
title Register
(*) -->"get login, pass\n and verification pass"
if "pass==verification pass?" then
  note left: show "passwords don't match" msg
  -left->[0]"get login, pass\n and verification pass"
else
  -->[1]"send password and login to server"
  if "sever recieved data in 3 seconds?" then
  note left: show "could not connect to server" msg
            -->[0]"get login, pass\n and verification pass"
  else
            if "server saved login and pass\n to database?" then
              note right: show "saving data error \n please try again" msg
    -->[0]"get login, pass\n and verification pass"
            else
                        -->[1](*)
            endif
  endif
endif
@enduml
```

Процес перегляду куплених та попередньо заброньованих квитків

```
@startuml
title Check purchesed and reserved tickets
(*) -->"send request to get\n bought and reserved tickets list"
if "server respond?" then
            note right: "could not connect to server" msg
            -left->[0]"send request to get\n bought and reserved tickets list"
else
            -->[1]"show recieved list"
            if "user vant to buy previously reserved ticket?" then
                        -->[0](*)

            else
   partition extend   {
     -->[1]"buy pteviously reserved tickets"
   }
                        -->(*)
            endif
endif
@enduml
```

Процес покупки попередньо заброньованих квитків

```
@startuml
title Buy pteviously reserved tickets
(*) -->"send request to get\n reserved tickets list"
if "server respond?" then
            note left: "could not connect to server" msg
            -left->[0]"send request to get\n reserved tickets list"
else
            -->[1]"show recieved list"
  -->"get a shopping list"
  -->"send a shopping list to server"
  if "server responded\n that purchase is done?" then
            note left: show "the transaction cannot be completed\n please try again" msg
    -->[0]"show recieved list"
  else
    -->[1]"show purchase is seccessful msg"
```

```
   -->(*)
 endif
@enduml
```

Процес перегляду доступних на потрібний потяг квитків

```
@startuml
title Check tickets availability
(*) -->"send request to server\n to get cities list"
if  "do we get cities list\n back from server?" then
        [0]if "Does user want to\n try connecting again?" then
                -->[0](*)
        else
                -->[1]"send request to server\n to get cities list"
        endif
else
        -->[1]"get departure"
        if "Does departure exist in the\n cities list?" then
                -->[0]"get departure"
        else
                -->[1]"get destination"
                if "Does destination exist in the\n cities list?" then
                        -->[0]"get destination"
                else
    -->[1]"get depart time"
                        -->"send departure, destination and depart time to server\n to get needed trains list"
                        if "do we get trains list\n back from server?" then
                                [0]if "Does user want to\n try connecting again?" then
                                        -->[0](*)
                                else
                                        -->[1]"send departure, destination and depart time to server\n to get needed
trains list"
                                endif
                        else
                                -->[1]"show recieved trains list"
                                -->"get selected by user train info"
                                -->"send train info to server"
                                if "do we get vagons list\n back from server" then
                                        [0]if "Does user want to\n try connecting again?" then
                                                -->[0](*)
                                        else
                                                -->[1]"send train info to server"
                                        endif
                                else
                                        -->[1]"show vagons list with info\n about aveliable places"
                                        -->"get selected by user vagon info"
                                        -->"send vagon info to server"
                                        if "do we get places in selected\n vagon list back from server?" then
                                                [0]if "Does user want to\n try connecting again?" then
                                                        -->[0](*)
                                                else
                                                        -->[1]"send vagon info to server"
                                                endif
                                        else
                                                -->[1]"get selected by user places info"
                                                -->"send places info to server"
                                                if "do we get places in selected\n vagon info back from server?" then
                                                        [0]if "Does user want to\n try connecting again?" then
                                                                -->[0](*)
                                                        else
                                                                -->[1]"send places info to server"
                                                        endif
                                                else
                                                        -->[1]"show price"
                                                        if "what does user want?"
                                                                -->[exit](*)
                                                        else
```

```
                                                    partition Reserve {
                                                        -->[reserve ticket]"reserve ticket"
                                                    }
                                                    -->(*)
                                                else
                                                    partition Buy {
                                                        -->[buy ticket]"buy ticket"
                                                    }
                                                    -->(*)
                                                endif
                                            endif
                                        endif
                                    endif
                                endif
                            endif
                        endif
                endif
        endif
endif
@enduml
```

Процес покупки доступних на потрібний потяг квитків

```
@startuml
title Buy ticket
(*)-->"send tickets list to server"
if "Do server responded that purchase is done?" then
        [0]if "Does user want to\n try connecting again?" then
                -->[0](*)
        else
                -->[1]"send tickets list to server"
        endif
else
        -->[1]"show purchase is good msg"
        -->(*)
endif
@enduml
```

Процес резервації доступних на потрібний потяг квитків

```
@startuml
title Reserve ticket
(*)-->"send tickets list to server"
if "Do server responded that reservation is done?" then
        [0]if "Does user want to\n try connecting again?" then
                -->[0](*)
        else
                -->[1]"send tickets list to server"
        endif
else
        -->[1]"show reservarion is good msg"
        -->(*)
endif
@enduml
```

Процес виходу з системи

```
@startuml
title Exit
(*)-->"user click the exit button"
  -->" ask does user really want to log out?"
  if"User really want to log out" then
    -->[0]"go to previous menu"
  else
    -->[1](*)
  endif
@enduml
```

Діаграми активності серверної частини та їх опис

Процес декодування отриманої від клієнта команди

```
@startuml
title Decoding command recieved from client
(*) --> "sent confirmation of data receiving to client "
-->"check code of recieved request"
if "what value does code have?" then
	partition UserVerification {
		-->[0]"verify user"
	}
	-->"send msg to client"
else
	partition AddingNewUser {
		-->[1]"add new user"
	}
	-->"send msg to client"
else
	partition FindingUserPurchases {
		-->[2]"find user purchases"
	}
	-->"send msg to client"
else
	partition FindTheCity {
		-->[3]"find the city"
	}
	-->"send msg to client"
else
	partition FindingNeededTrains {
		-->[4]"find needed trains"
	}
	-->"send msg to client"
else
	partition FindTheVagons {
		-->[5]"find the vagons"
	}
	-->"send msg to client"
else
	partition FindThePlace {
		-->[6]"find the place"
	}
	-->"send msg to client"
endif
-->(*)
@enduml
```

Процес верифікації користувача

```
@startuml
title User verification
(*) --> open "users and passes" database
if "user name was found?" then
  -->[0] create negative respond
  -->(*)
else
  [1]if "recieved pass == pass in datavase?" then
    -->[0] "create negative respond"
  else
    -->[1] "create positive respond"
    -->(*)
  endif
endif

@enduml
```

Процес додавання нового користувача

```
@startuml
title Adding new users
(*) --> open "users and passes" database
if "user name was found?" then
  -->[1] create negative respond
```

```
  -->(*)
else
  -->[0]"add username and pass to database"
  -->"create positive respond"
  -->(*)
endif

@enduml
```

Процес пошуку попередньо заброньованих та куплених квиктів

```
@startuml
title Finding user purchases
(*) --> open "users purchases" database
-->"find user in database"
if "user name was found?" then
  -->[0] create negative respond
  -->(*)
else
  -->[1]"create purchases list"
  -->(*)
endif

@enduml
```

Процес пошуку міст

```
@startuml
title Find the city
(*) --> open "cities" database
-->"add every unique city to array"
-->"create respond"
-->(*)

@enduml
```

Процес пошуку потрібного потягу

```
@startuml
title Find needed trains
(*) --> open "cities and trains" database
-->"find departure city"
if "Departure city was found?" then
  -->[0]"create negative respond"
else
  -->[1]"find destination city"
  if "Destination city was found?" then
    -->[0]"create negative respond"
    -->(*)
  else
    -->[1]"create list with all possible\n trains from departure to\n destination that depart in needed day"
    -->"create positive respond"
  endif
endif
-->(*)

@enduml
```

Процес пошуку вагону

```
@startuml
title Find the vagons
(*) --> open "vagons" database
-->"find train vagons \nand free spaces in them"
if "vagons were found?" then
  -->[0]"create negative respond"
  -->(*)
else
  -->[1]"create vagons and places list"
  -->(*)
endif
@enduml
```

Процес пошуку місця

```
@startuml
title Find the place
(*) --> open "places" database
-->"find places in vagon"
if "placeswere found?" then
  -->[0]"create negative respond"
  -->(*)
else
  -->[1]"create places list"
  -->(*)
endif
@enduml
```