Fakulta informatiky a informačných technológií
Slovenská technická univerzita

Martin Bopkó

Modelovanie softvéru

Management system for travelers and traveling groups

Travel simply

Thursday: 16:00 – 18:00
MSc. Muhammad Nasim Bahar, M.Sc

System for travelers and traveling groups

Travel simply

Project topic: Trip Planning

These days longer trips require lot of management. For somebody it is complicated to manage his/her time or get his plans in order. Not to talk about to have all important documents in one place or just get the necessary information about the place he/she wants to visit. This system would be responsible for managing and synchronizing travels for groups or individuals.

Project objective

System provides the options such as planning, organizing the whole journey. The traveler just needs to register and create an account to use the program. It enables to save the necessary files such as tickets, reservations ... etc. to have all documents in one place.

The travel plan is presented and organized as a workflow. Each day would be separately planned. The traveler can add and delete places of interest, hotels, and the time when he/she plans to visit these places. If the traveler wants to visit a destination which is stored in the system, the system will give suggestions, what can be interesting to visit or the required documents, vaccines etc..

As a special option, there is also chance to share the trip with others. The traveler just needs to create an invitation manually or find their profile and choose the invite option. Next traveler adjusts the level of access of the invited one, whether he/she can also plan the shared trip, or just has the right to see it. This can be beneficial for groups. The member can upload their expanses from the trip, so they knew how much they owe each other.

If one visits a place or a spot which comes in the way, the traveler has the option to save the place to his favorites. The traveler also can review the visited places or the trips of others, so the system could suggest them and show the review to other travelers. The whole trip can be set as public, if others wanted to copy the trip, so they would have a complete plan with estimate prices, as well as review trips of others.

Content

The following are the use cases that are part of the system. I'm using **Cockburn** notation. UC, activity and sequence diagrams are described in EA in the notes section.

UC01 Enter the system

Actors: User, System

Preconditions: User wants to enter the system, either by login, or registration.

Post conditions: User entered the system.

Main scenario:

1. User wants to enter the system.
2. User opens the application.
3. User chooses how to enter.
4. User fills data and confirms.
5. System validates data.
6. User entered the system.

Alternative scenario

After step	**Continues at**	**Alt**
3a	4	**Alt1:** User wants to register. System shows registration formular. User fills it. Account is created.
3b	4	**Alt2:** User wants to login. Fills login data.

Exception scenario

After step	**Continues at**	**EXC1:** User entered invalid login data. User is not logged in.
5a	END	
5b	END	

		EXC2: User didn't fill all necessary fields.

UC02 Invite traveler

Actors: Traveler, Invited traveler, System

Pre - conditions: Traveler is registered and already created a trip plan. Traveler wants to invite other travelers to join the trip.

Post - conditions: Invitation is sent to the invited traveler.

Main scenario:

1. Traveler chooses to invite traveling partners.
2. Traveler creates invitation.
3. Traveler adds email or username of the invited traveler.
4. Traveler chooses to which trip will be the invited attached to.
5. Traveler adjusts level of access of the invited traveler, whether the invited will be collaborator (can adjust the plan) or spectator (just sees the plan).
6. Traveler confirms invitation.
7. **UC04 Send invitation**.

Alternative scenario

After step	**Continues at**	**ALT1:** Traveler creates invitation manually.
2a	3	
2b	4	**ALT2: UC05** Search for user profile

Exception scenario

After step	**Continues at**	**EXC1:** Traveler cancels invitation. Invitation is not sent.
5a	END	

UC03 Search for user

Actors: Traveler, system

Preconditions: Traveler wants to find profile of another user.

Post conditions: System shows the found profile.

Main scenario:

1. Traveler opens search.
2. Traveler writes the name he is looking for.
3. Confirms search.
4. System shows the found profile.

Alternative scenario: -

Exception scenario:

After step 3a	**Continues at** END	**EXC1**: Username doesn't exist. System informs the traveler.
3b	2	**EXC2**: Traveler didn't fill the username. System warns the Traveler.

UC04 Send invitation

Actors: System

Preconditions: Traveler has created and confirmed to send invitation

Post conditions: Invitation is sent

Main scenario:

1. Invitation is confirmed
2. System validates the invitation
3. System searches for the invited user.
4. System notifies the sender, that invitation has been sent.
5. System sends the invitation

Exception scenario

After step	**Continues at**	**EXC1:**
3a	END	**EXC1:** System doesn't find searched user. Invitation is not sent.
3a	END	**EXC2:** Required data is missing. Invitation is not sent.

UC05 Response invitation

Actors: Invited traveler

Preconditions: User got invitation.

Post conditions: User is added to the trip.

Main scenario:

1. Invited traveler gets notification
2. Invited traveler opens notifications
3. Invited traveler accepts invitation.
4. Invited traveler is added to trip.

Exception scenario

After step	Continues at	EXC1
3a	END	**EXC1:** Invited traveler refuses the invitation. Invited traveler is not added to the trip.

UC06 Create trip

Actors: Traveler, System

Preconditions: Traveler is logged in and wants to create new trip.

Post conditions: Trip is created.

Main scenario:

1. Traveler chooses to create new trip and clicks the new travel button.
2. **UC07 Set plan**.
3. **UC08 Show information about the destination**.
4. Traveler confirms the trip.
5. Trip is created.

Alternative scenario: -

Exception scenario:

After step 4a	**Continues at** END	**EXC1:** Traveler cancels the trip. Trip is not created.

UC07 Set plan

Actors: Traveler

Preconditions: Traveler choses to create trip.

Post conditions: Duration of trip, destination is set.

Main scenario:

1. Traveler sets the starting date of the trip.
2. Traveler sets the ending date of the trip.
3. Traveler sets destination.
4. Confirms plan.
5. Plan is created.

Alternative scenario:

Exception scenario: -

After step **4a**	**Continues at** END	**EXC1:** Traveler didn't fill the required data.
4b	END	**EXC2:** Destination is not valid.

UC08 Show information about the destination

Actors: System

Preconditions: Base trip plan is set. Destination and duration is set.

Post conditions: Information about the destination is showed.

Main scenario:

1. System searches if the destination is already stored.
2. System shows mandatory documents what the traveler must have, makes suggestions.
3. System shows most visited places near to destination.

Alternative scenario:

Exception scenario:

UC09 Adjust day

Actors:

Preconditions: Trip is created on traveler's profile.

Post conditions: Plans of the exact day is changed.

Main scenario:

1. Trip member (traveler or invited user set as collaborator) clicks on the trip.
2. System shows list of days of the trip.
3. Member chooses which day he wants to change.
4. Member adjusts the plan.
5. Member adds place of interest.
6. Member sets duration.
7. Confirms.
8. System actualizes the day.

Extension point: Add place of interest

1. Traveler names the place he wants to visit.
2. System finds the place.
3. Traveler saves the place.

After step	**Continues at**	**ALT1:** Member adds place of interest or destination.
6a	5	

UC010 Save expanses

Actors: traveler, invited traveler, system

Preconditions: Travelers want to split their expanses.

Post conditions: System calculates how much they each other.

Main scenario:

1. User fills for what he has paid for.
2. User fills the sum.
3. User saves the expanse.
4. System splits the expanses and calculates how much the members owe each other.

Alternative scenario: -

Exception scenario:

After step	**Continues at**	**EXC1:** Traveler inputs wrong data type. Expanse cannot be saved.
3a	END	

UC011 Transfer money

Actors: User

Preconditions: Users owe money to each other. User wants to transfer money.

Post conditions: Money is transferred

Main scenario:

1. Expanses are calculated
2. User decides to transfer money
3. User fills transfer data.
4. User confirms transfer.
5. **UC12** Approve transaction.

Alternative scenario

After step: 3a	**Continues from**: 4	**ALT1:** User fills transaction data manually
3b	4	**ALT2:** **UC011** Quick transfer

Exception scenario

After step 5a 4a	**Continues at** END END	**EXC1:** UC Approve transaction fails, so transaction is not completed. **EXC2** Some mandatory fields were let empty. System warns the user.

UC012 Quick transfer

Actors: User

Preconditions: User is transferring money

Post conditions: Transaction data are filled.

Main scenario:

1. User chooses quick transfer.
2. System starts QR code scanner.
3. User scans QR code.
4. All transfer data are filled automatically

Alternative scenario -

Exception scenario -

After step **3a**	**Continues at** **2**	**EXC1:** QR code scanning fails. QR is not scanned, data are not filled.

UC013 Approve Transaction

Actors: Bank

Preconditions: Transaction data is confirmed.

Pos conditions: Transaction is approved and sent.

Main scenario:

1. Bank gets transaction data of the traveler.
2. Bank validates transaction data.
3. Bank approves transaction.
4. Money is transferred.

Exception scenarios

After step	Continues at	EXC1: Validation fails, transaction is not approved.
2A	END	

UC014 Save files

Actors: User
Preconditions: User is logged in, and there are files he wants to save.
Post conditions: Files are saved in his account.
Main scenario:

1. User clicks save file button.
2. System shows file types that can be saved.
3. User chooses what he wants to save.
4. User confirms.
5. System validates file.
6. Files are saved in the profile.

Alternative scenario

After step	Continues at	Alt1: User saves document. Alt2: User saves photo. Alt3: User saves QR code.
3a	4	
3b	4	
3c	4	

Exception scenario

After step	Continues at	EXC1: File is too big and cannot be saved.
3a	END	

UC015 Save place as favorite

Actors: User

Preconditions: User is logged in and wants to save a place to the list of favorites.

Post conditions: Place is saved.

Main scenario:

1. User clicks the save favorite button.
2. System shows the fields.
3. User fills the fields.
4. System saves the place to the list.

Alternative scenario: -

Exception scenario: -

After step	**Continues at**	**EXC1:** User cancels the saving.
2a	END	

Code explanation

First screen is the login screen. There are 3 test accounts, which can be logged in to test the application.

Username1: User1

Password1: 1234

Username2: User2

Password2: 12345

Username3: User3

Password3: 123456

I created a class for the data structures called Storage, which will hold the objects I work with. This class implements the Search interface.

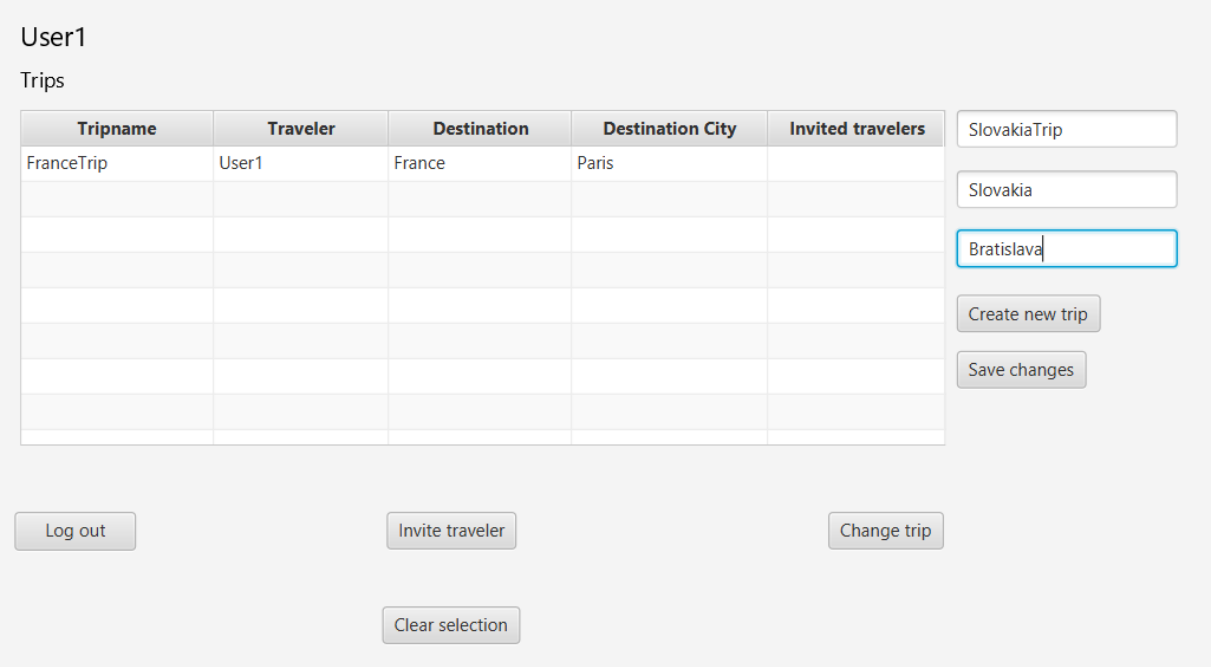
Besides users, I also defined their trips.

```
User u1 = new User( username: "User1",  email: "testuser@gmail.com",  ID: "u101",  password: "1234");
User u2 = new User( username: "User2",  email: "testuser2@gmail.com",  ID: "u202",  password: "12345");
User u3 = new User( username: "User3",  email: "testuser3@gmail.com",  ID: "u303",  password: "123456");

City c1 = new City( name: "Paris",  country: "France",  info: "Eifel tower, Louvre");
City c2 = new City( name: "Prague",  country: "Czech republic",  info: "Karluv most, Orloj, Vaclavovo namesti");
City c3 = new City( name: "Rome",  country: "Italy",  info: "Colosseum, Forum Romanum, Angels Castle, Vatican");

Country country1 = new Country( name: "France",  info: "No required vaccination");
Country country2 = new Country( name: "Czech republic",  info: "No required vaccination");
Country country3 = new Country( name: "Italy",  info: "No required vaccination");

Trip trip1 = new Trip( traveler: "User1",  name: "FranceTrip",country1.getName(), c1.getName());
Trip trip2 = new Trip( traveler: "User2",  name: "ItalyTrip",country3.getName(), c3.getName());
```

Cities hold the name, country, and an information parameter. This would represent the show information UC I defined in my project. Similar approach is in the Country objects.

Trips are created by Users. The traveler parameter defines who has created the trip, who is the owner. Then the name of the trip, the destination country and destination city.

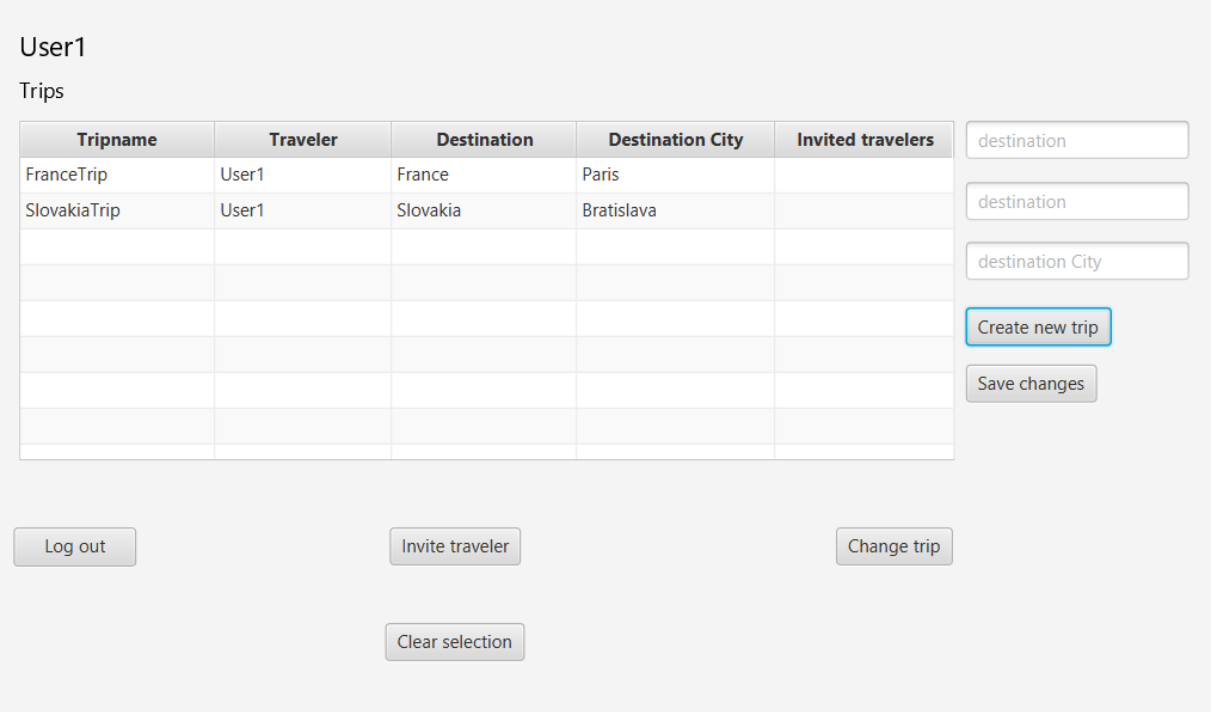
After login, this screen is loaded.



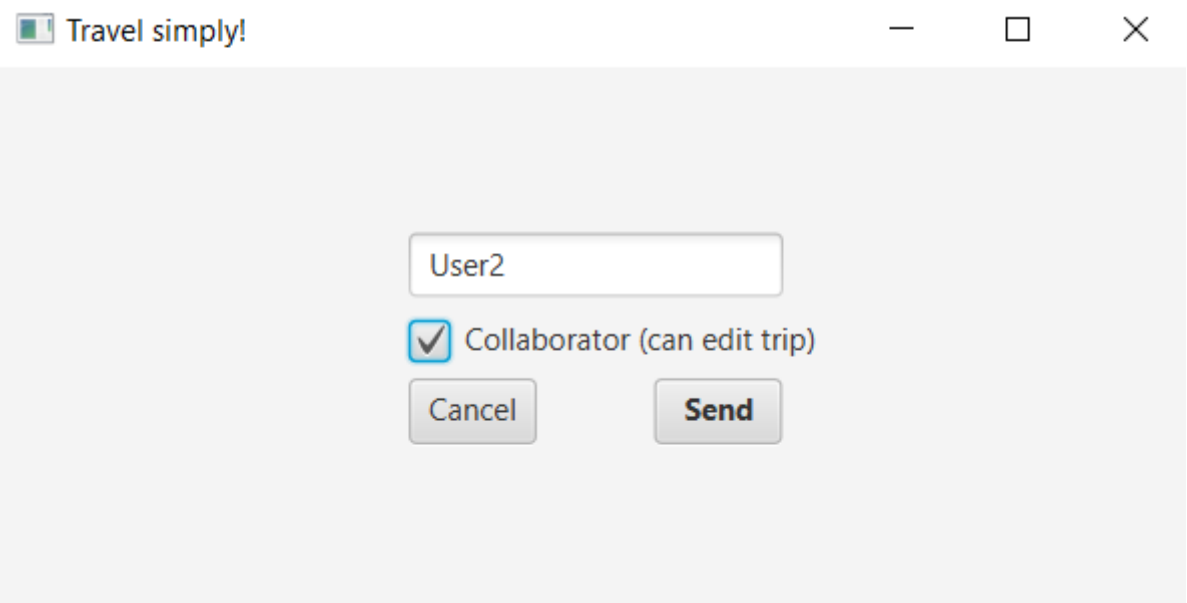
Here User1 is logged in. The table holds all the trips he has created, or was invited in. As I created in the test scenario, User1 has a default trip set, the FranceTrip. Next to the table are text fields if user wanted to create a new trip.

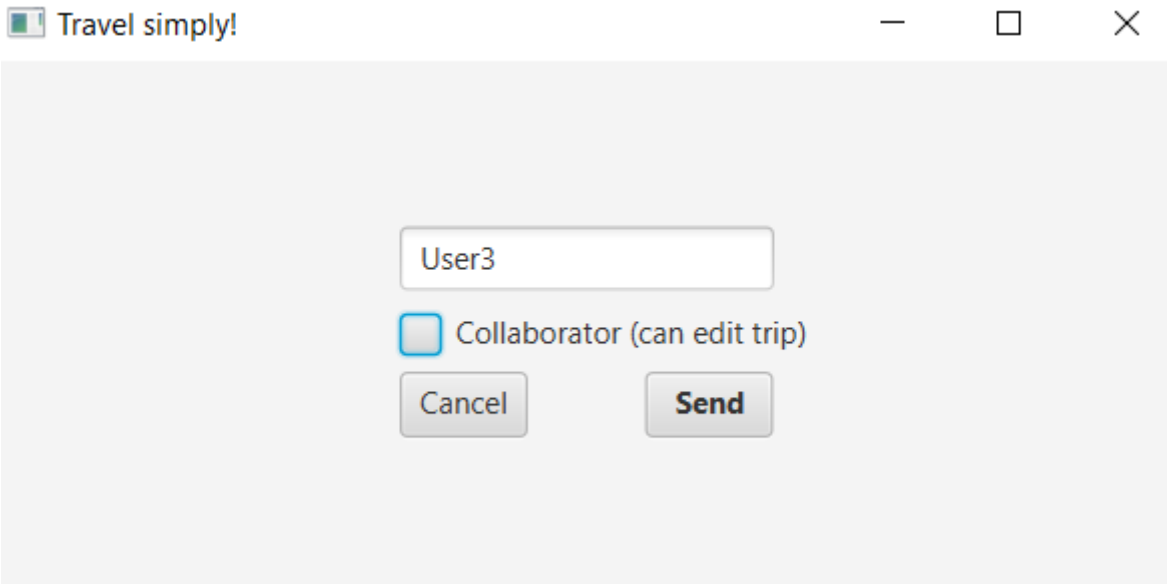
First field is for the name of the trip, second for destination country and third for the destination city.

After clicking the Create new trip button, trip will be created and saved into the table.

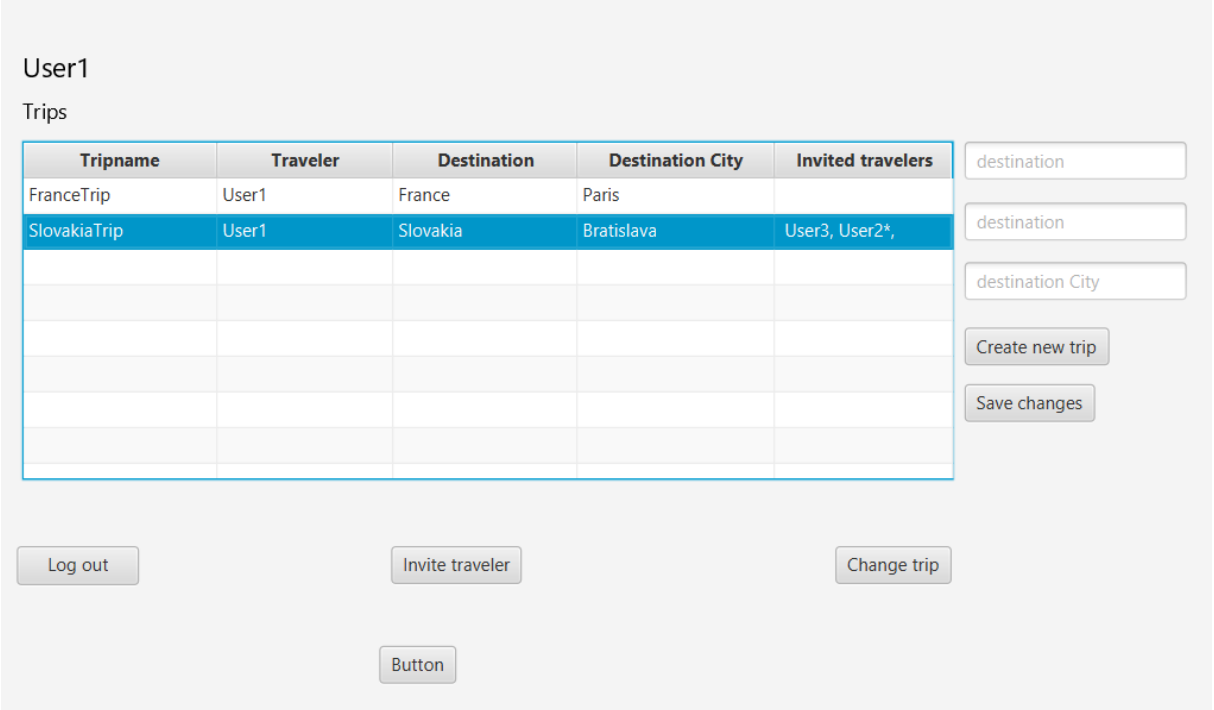


In order to invite other user to a trip, user first has to click on one of the trips/records. After that user clicks the Invite traveler button new screen is opened:



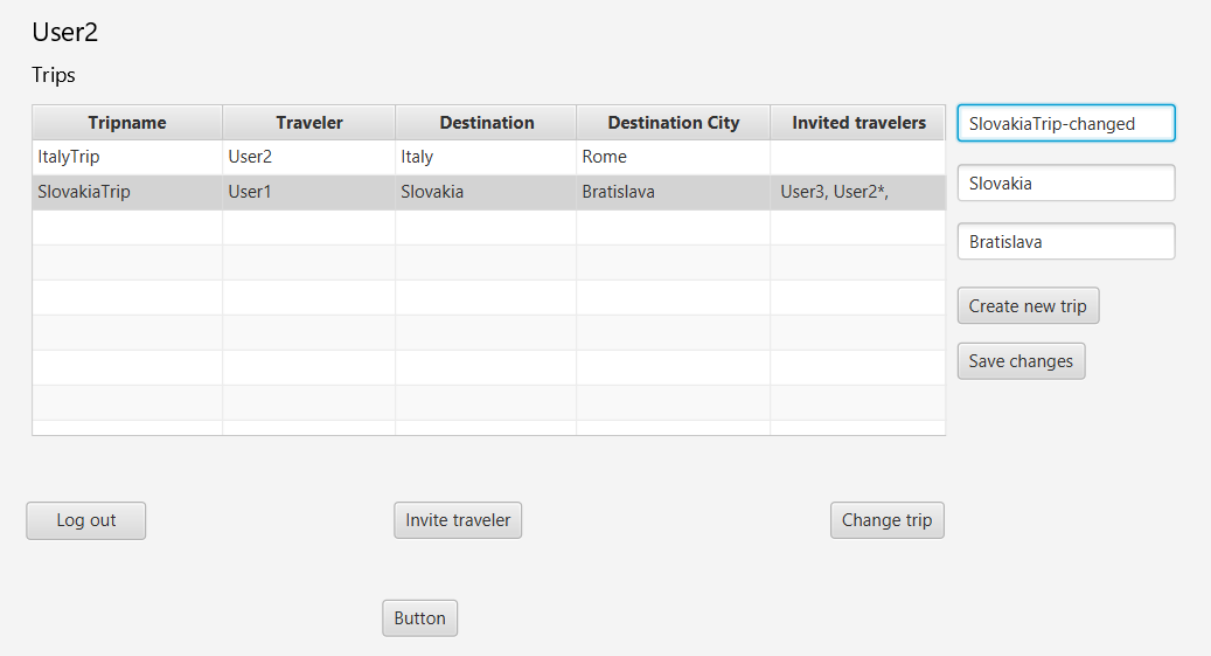


It’s a simple implementation of the invitation method I defined in the project. Here user just has to type the other existing user’s name : (User2 or User3) , next he checks the access checkbox. (If it is checked, than the invited will be able to change the tripname, destination) and click the Send button and the invitation User will be added to the trip.

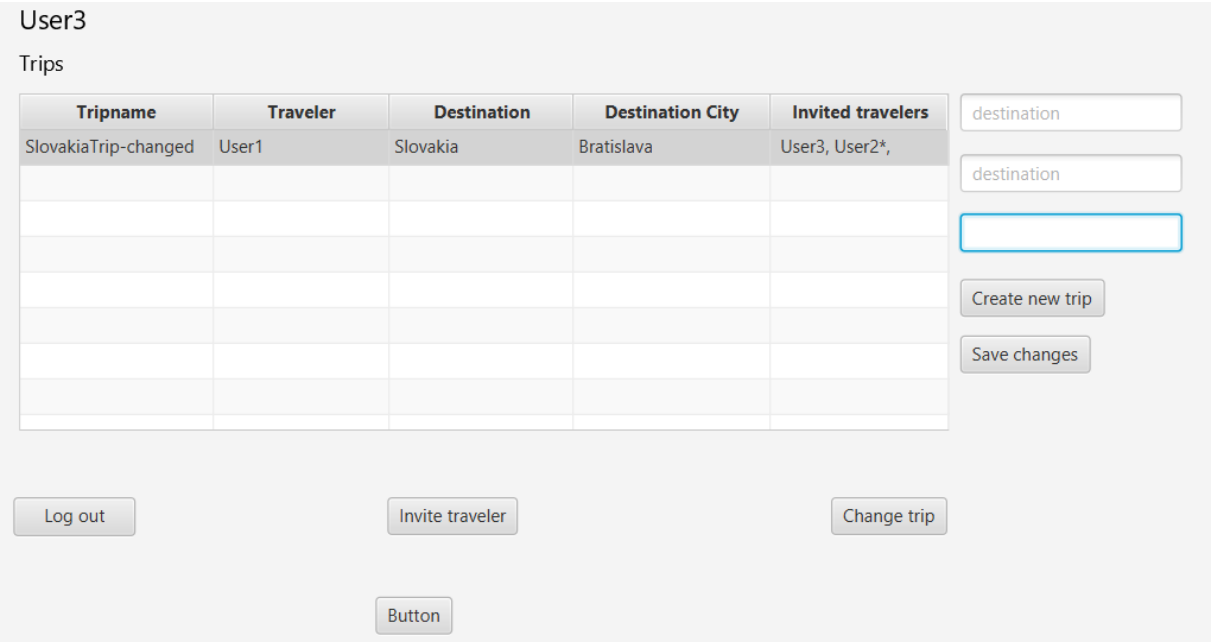


Here we see that invited users are in the last column. Collaborator are also signed with an * sign. (User2).

Next user can log out and login with the account he has sent the invitation for.

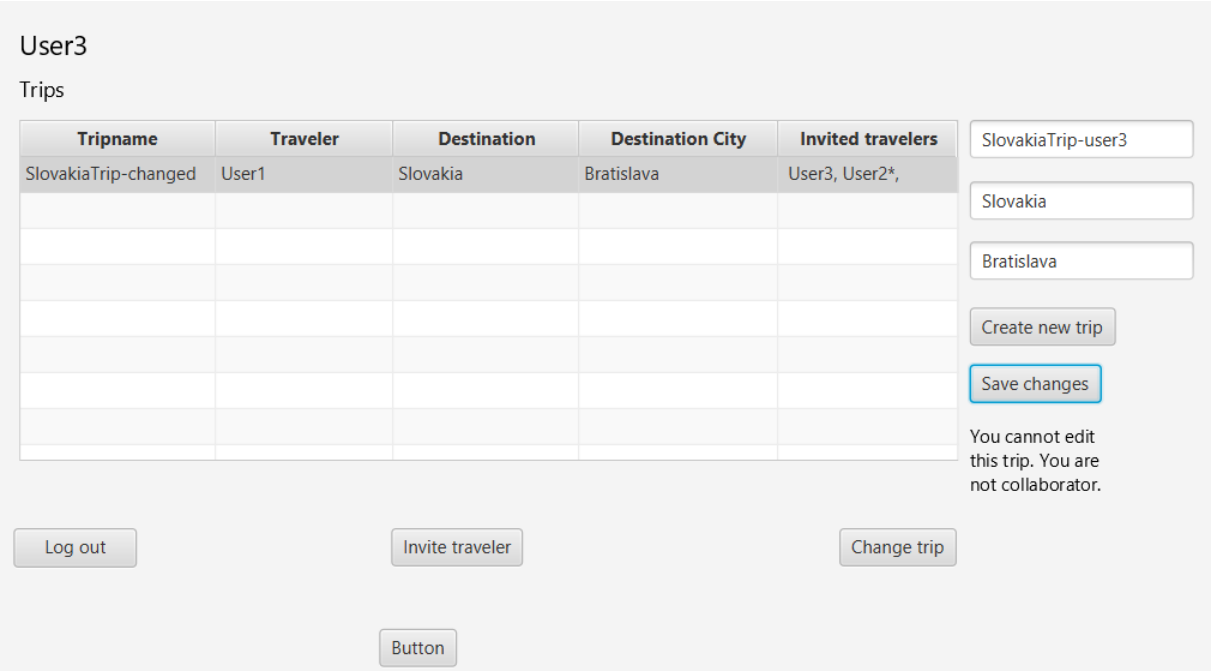


Here we see SlovakiaTrip in the table of User2, so he got the invitation. ItalyTrip was set in the beginning. Here I also selected the Slovakia trip, then added it to the textfields with Change trip button. I edited the tripname and clicked the Save changes button.



After User3 login we can see that User2 could change the trip as the tripname is SlovakiaTrip-changed.

However, when User3 tries to change it, it won't change, as he isn't collaborator.



Code was written in Java 18.

OCL

OCL expressions are in package Class diagrams->Whole class diagram.

context Invitation::Invitation InvitationManager.accept(invitation)
post: invitation.trip.companionIDlist.size() = invitation.trip.companionIDlist.size()@pre+1

context Invitation::Invitation InvitationManager.sendInvitation(reciever)
pre: self.reciever != null
post: self.reciever.invitations.size() =self.reciever.invitations.size()@pre+1

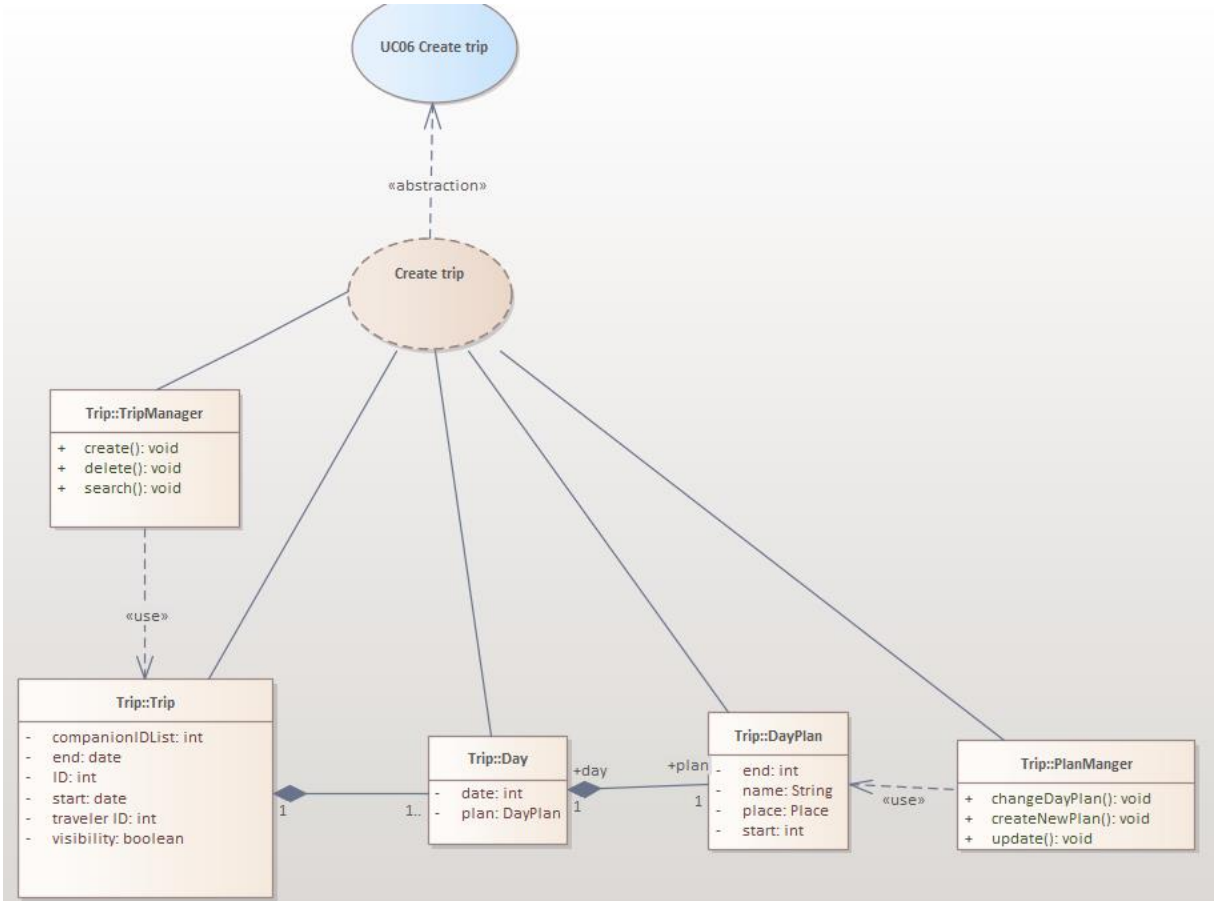
context: Trip
inv: self.start < self.end

Found message

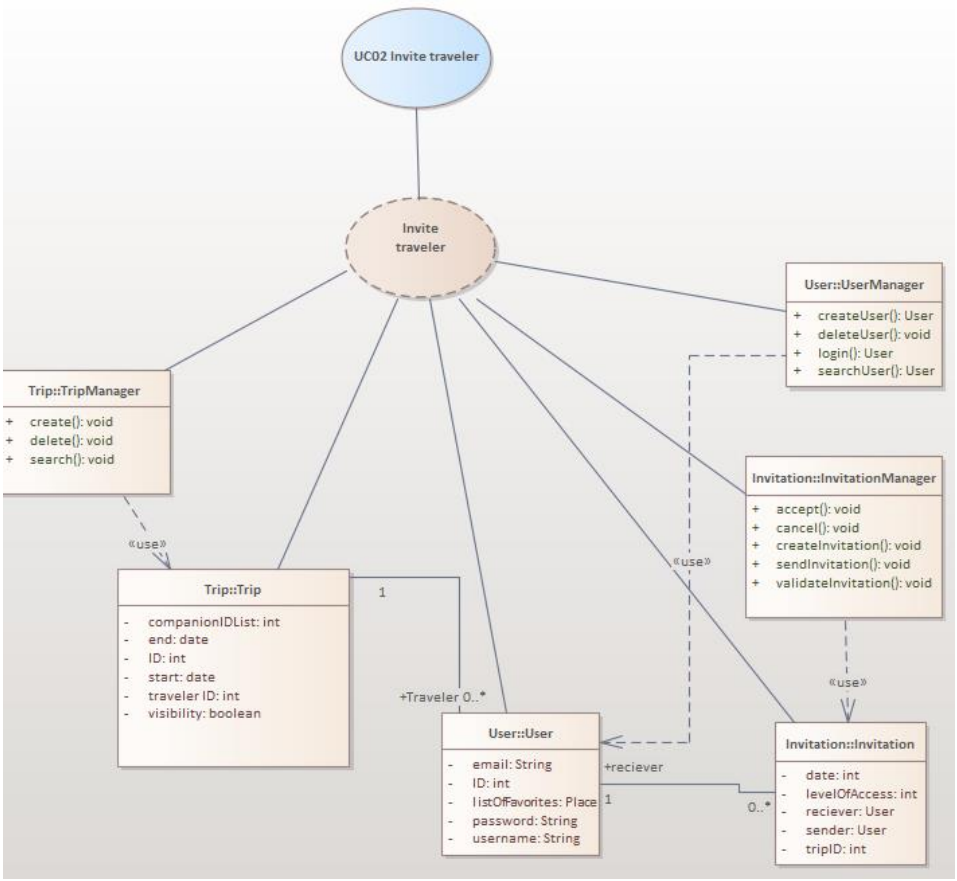
Diagrams are in EA file, in package classDiagram -> sequenceDiagram – found message.

Collaboration diagram - Collaboration of classes

Creating a trip requires the collaboration of these classes.



Sending invitation.


UC02 Invite traveler
Invite traveler
User::UserManager
+ createUser(): User
+ deleteUser(): void
+ login(): User
+ searchUser(): User
Trip::TripManager
+ create(): void
+ delete(): void
+ search(): void
Invitation::InvitationManager
+ accept(): void
+ cancel(): void
+ createInvitation(): void
+ sendInvitation(): void
+ validateInvitation(): void
«use»
«use»
«use»
Trip::Trip
- companionIDList: int
- end: date
- ID: int
- start: date
- traveler ID: int
- visibility: boolean
1
+Traveler 0..*
User::User
- email: String
- ID: int
- listOfFavorites: Place
- password: String
- username: String
+reciever
1
0..*
Invitation::Invitation
- date: int
- levelOfAccess: int
- reciever: User
- sender: User
- tripID: int