**A Community for gathering people whose destination is same**

**Description**

When you travel with little amount of people, you are hesitant to take a taxi or other transport. It’s not a big problem if you are skillful traveler or the rich, but most of people feel more comfortable if they find someone move with. With your companion, you will not only save money, but also be able to start your travel more easily. Also, “sharing” is becoming the trend of era. It is not just sharing the memory or information. It can divide money and share skill.

The objective of this project is to make travel more comfortable and reduce the travel expenses. Unlike the travel mate matching applications that have been released before, it does not pursue emotional exchange. It is because some security issues are important these days. It connects all the routes and saves it to protect people from criminal by getting the information from airlines, some accommodations and gathering the location information of users only until they arrive to destination.

Users can meet and make an appointment in this app. They can use it as a messenger app, and for the safety, it never provide each others’ important information. There are options for meeting with same sex and the number of companion.

\*Ensure that all material is referenced correctly.

**Requirements Engineering**

**Feasibility Study**

1. Identify at least 4 examples of E-Learning systems in the marketplace. Briefly describe each of them. Reference the URL for each. (in Windows, Android, iOS, …)? *(min 100 words)*

1 serentrip : platform for meeting travel buddy and local people.

<https://play.google.com/store/apps/details?id=cool.likecrazy.at&hl=en>

2 travelpal : provides all kinds of information of destinations like climate, currency or anything. And it supports to make travel buddy. Also it provides translation service.

<https://play.google.com/store/apps/details?id=com.hellopal.travel.android&hl=en_US>

3 tinder : Its main purpose is not for trip, but for meeting. It has the similar function, “Passport”. It makes people meet near travel site.

<https://play.google.com/store/apps/details?id=com.tinder&hl=en>

4 Mochileros Amino : community for backpackers. They share their opinions, tips, and discuss about being backpacker.

<https://play.google.com/store/apps/details?id=com.narvii.amino.x175875997&hl=en_US>

5 Travel Buddy : helps to make a plan for trip, provides various information of destination. Also same with other things, give the chance to meet travel mate.

<https://play.google.com/store/apps/details?id=com.beatravelbuddy.travelbuddy&hl=en>

1. Identify the main system features and services provided in travel community applications. Consider the existing systems and the services they provide to users.

[Reference and copy the URLs] *(min 200 words)*

Most apps use chat system to provide a function that makes them meet and talk to travel buddies or local friends. Also, in case of providing local information, they use maps API. Besides, some have database for accommodations that enables to reserve hotels or hostels through the application.

On the basis of these, main system will have chat system. Also, for meeting with purpose of project, this app use database for airports, flights and, accommodations. Only users who register their flight information are able to use specific board. For supporting to find ways, map API will be provided. Also it will be linked to local transportation to make people easier to move. Payment system will be included because it is easy to use this feature when the payment system is supported.

1. Describe a new type of travel community and how it might operate. Consider existing systems that provide services to users. Use the internet to identify

My new type of travel community is different from other ordinary matching applications. It provides function that finds companions only to destinations. In other apps, they support to exchange some emotional experiences, make a new friend. But with this app, you just share a taxi, help each other to destinations, find a way to goal with your companion. That’s all. It is all up to you like greeting each other, hanging out with and making an appointment to eat outside.

1. In what ways do travellers currently meet other people? *(min 100 words)*

Travelers meet people or local residents with using other apps or just going out some communities like pub or club. They meet new people there and make relationship. Other way, people meet on apps, talk to each other, and make an appointment. Or they meet at their accommodations. In hostels or guest houses’ case, there are many solo travelers. Because their travel plans are similar to each other, with sharing their room and eating outside with, they become friends. In other way, people just travel alone or just hang out with their friends who were originally travelling together. In this case, people do not meet other people.

1. Who are the stakeholders? Would this app affect them positively or negatively? *(min 30 words)*

Users, who travel alone or want to save the money, are going to get profit from this app. It can give positive effect to them by matching friends, but it could affect negatively. Because meeting new person can be very dangerous and some people will use it as cruel purpose.

1. What other research would be necessary to ascertain feasibility e.g., ownership of smartphones/tablets…? *(min 100 words)*

It is necessary to figure out travelers’ needs and thought about “traveling with other people”. And it needs to gather flight information of all airports and airlines. With providing that, people could find companion securely. It could be helpful if this application has connections with local taxi or public transportations. For this, research about preferred transportations is also important. Survey about how to meet other people will be helpful, too.

1. Make an initial list of **functional** and **non-functional** requirements.

Functional list : chatting or bulletin board system, map, booking system, payment system, profile, options for preferred sex and number of companions, transportations.

Non-functional list :

**Requirements Elicitation**

1. Could ethnographic methods be used in this case study? If so, in what way? *(min 30 words)*

I think so. Because there are so many other different societies, and different norms. It does not help directly to this case, but it will be helpful to users’ action. According to environment, people should accept to the change. Ethnographic methods will be very helpful to that.

1. Identify a significant stakeholder(s), which will be **interview**ed to get more information on the intended product. Justify your choice of stakeholder(s). Do up an interview plan and pre-prepare approximately 10 questions.
2. Users(Traveler) : actual user of app is the most significant stakeholder
   1. Have you traveled alone?
   2. How many times do you go trip alone?
   3. What transportation do you prefer to use when you move from airport to accommodation?
   4. Did you feel transportation fee is expensive?
   5. If you can find companion who could go with only airport to destination, will you go with?
   6. Have you used any apps about travel before?
   7. Do you feel any uncomfortable things from it? If so, tell me about it.
3. Local merchants : By providing their information using app, they can get more customers.
   1. Have you used any linked apps supporting your business before?
   2. What kinds of apps do you use?
   3. Do you think you can get any profit from those apps?
   4. If your business advertisement displays on traveling app, is it helpful to you and your business?
   5. When number of travelers increases, do your business profit also increase?
   6. If you are taxi driver, what do you want the traveling app to provide?
   7. If you are hotel/hostel/guest house/Airbnb host, do you prefer to customers come with same time or other time?
4. Identify a significant group of stakeholders, which will receive **questionnaires**. Justify your choice of stakeholders. The questionnaire that you create should have approximately 10 questions.

# Requirements Analysis

1. Use the use case template to analyse the proposed system

Draw an initial *use-case diagram* with supporting scenario description for this app (possibly using *StarUML* for the diagram). The first iteration of the use-case diagram can consist of a single overall use case with supporting main flow and 2 or 3 alternative flows.

Community App

Users

(Travelers)

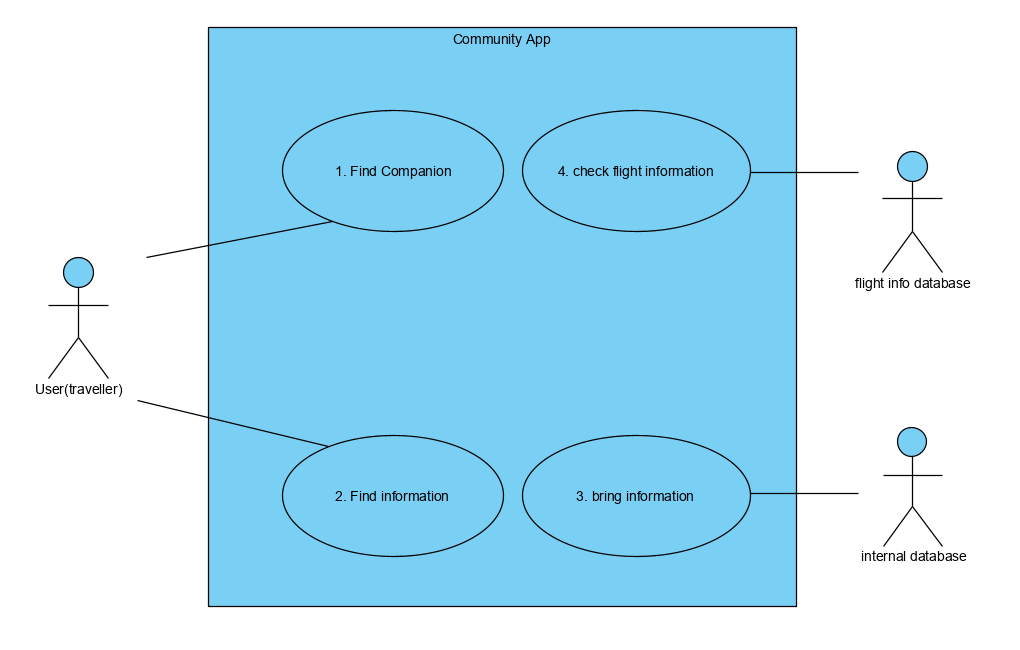
The use case description is developed from analysing the description of the use case. This is the statement of the goal of the use case.

For the first iteration this will be a description of the how the system operates.

Use Cases focus on functional requirements and specific system behaviour.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USE CASE** | | <number>  0 | <Name of Use Case>  <Using Service> | |
| **Description of Goal in Context** | | Users open their app and uses system. | | |
| **Preconditions** | | <what we expect is already the state of the system>  -Users turn on system. | | |
| **Post Conditions, Success End Condition** | | <the state of the system upon successful completion>  -Users can use other functions. | | |
| **DESCRIPTION** | | < The use case description is a paragraph identifying behaviour, it comes from the requirements gathering> | | |
| **Main Flow** | | | | |
| **Step** | **Action** | | | **Alternate** |
| 0.1 | Users turn on the app. | | |  |
| 0.2 | **Users click “Register” button.** | | | AF 0.2 |
| 0.3 | System checks information. | | | EF 0.3 |
| 0.4 | Users get authority to access. | | |  |
|  | | | | |
| **EXCEPTIONS or ERROR Flow Description** | | | | |
| **Step** | **Branching Action**  < Exception number m of Use Case n> | | | **Alternate** |
| 0.3 | <If information is wrong> | | |  |
| 0.3.1 | System indicates that registered information is wrong. | | |  |
| 0.3.2 | System request to correct information | | |  |
|  |  | | |  |
| **ALTERNATIVE or VARIATION Flow Description** | | | | |
| **Step** | **Branching Action** | | | **Alternate** |
| 0.2 | <Users don’t choose to register > | | |  |
| 0.2.1 | Indicate users cannot use service. | | |  |
| 0.2.2 | End Use Case | | |  |

1. Draw a second iteration in a separate word report consisting of 4 or 5 use cases. Each use case requires a use case narrative describing the scenario analysis. Each use case should have2 or 3 exception or alternative flows.



Users

(Travelers)

Community App

The use case description is developed from analysing the description of the use case. This is the statement of the goal of the use case.

For the first iteration this will be a description of the how the system operates.

Use Cases focus on functional requirements and specific system behaviour.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USE CASE** | | <number>  1 | <Name of Use Case>  <Find companion> | |
| **Description of Goal in Context** | | <By designating users’ own departure point, system gives user authority to access specific server. For example, if user1 uses one of flight from Seoul to Paris, she/he can register her/his flight on server, and only people who register their flight can access to server.> | | |
| **Preconditions** | | <what we expect is already the state of the system>  -flight information/location information should already exist on the server, app itself.  -users register their information. | | |
| **Post Conditions, Success End Condition** | | <the state of the system upon successful completion>  Users are able to use any space with an authority. | | |
| **DESCRIPTION** | | < The use case description is a paragraph identifying behaviour, it comes from the requirements gathering>  This use case exists for giving right authority and filtering right users. From this use case, people could meet others who depart from same place. | | |
| **Main Flow** | | | | |
| **Step** | **Action** | | | **Alternate** |
| 1.1 | Users turn on the app. | | |  |
| 1.2 | **Users click “Find Companion” button.** | | | EF 1.2 |
| 1.3 | System checks information. | | | EF 1.3 |
| 1.4 | Users get authority to access. | | |  |
|  | | | | |
| **EXCEPTIONS or ERROR Flow Description** | | | | |
| **Step** | **Branching Action**  < Exception number m of Use Case n> | | | **Alternate** |
| 1.2 | < If Users’ information is out-of-date>  System alerts that information is out-of-date. | | |  |
| 1.2.1 | System asks to register new information. | | | AF 1.2 |
| 1.2.2 | Users put their information. | | |  |
| 1.2.3 | Go to MF 1.3 | | |  |
| 1.3 | <If information is wrong> | | |  |
| 1.3.1 | System indicates that registered information is wrong. | | |  |
| 1.3.2 | System request to correct information | | | AF 1.2 |
| 1.3.3 | Go to MF 1.3 | | |  |
|  | | | | |
| **ALTERNATIVE or VARIATION Flow Description** | | | | |
| **Step** | **Branching Action** | | | **Alternate** |
| 1.2 | <Users don’t choose to register > | | |  |
| 1.2.1 | Indicate users cannot use service. | | |  |
| 1.2.2 | End Use Case | | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USE CASE** | | <number>  2 | <Name of Use Case>  <Find Information> | |
| **Description of Goal in Context** | | <Users could get some information about their travel destination like landmark, restaurants or weather. > | | |
| **Preconditions** | | <what we expect is already the state of the system>  -users click ‘information’ button  -system has destinations’ information. | | |
| **Post Conditions, Success End Condition** | | <the state of the system upon successful completion>  Users could get information of destination. | | |
| **DESCRIPTION** | | < The use case description is a paragraph identifying behaviour, it comes from the requirements gathering>  Users prefer to see information from one platform. They feel more comfortable when they could find useful information in easy way. This section meets their needs. | | |
| **Main Flow** | | | | |
| **Step** | **Action** | | | **Alternate** |
| 2.1 | Users click “INFORMATION” button | | |  |
| 2.2 | System shows useful information like weather, festivals, security or anything. | | | EF 2.2 |
|  | | | | |
| **EXCEPTIONS or ERROR Flow Description** | | | | |
| **Step** | **Branching Action**  < Exception number m of Use Case n> | | | **Alternate** |
| 2.2 | < If users don’t put information about their destination>  System gives choices to put their destination information or show random information. | | | AF 2.2 |
| 2.2.1 | System shows random destinations’ information. | | |  |
| 2.2.1 | Go to MF 2.2 | | |  |
|  | | | | |
| **ALTERNATIVE or VARIATION Flow Description** | | | | |
| **Step** | **Branching Action** | | | **Alternate** |
| 2.2 | <Users choose to put destination information> | | |  |
| 2.2.1 | Users put their information. | | |  |
| 2.2.2 | Go to MF 2.3 | | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USE CASE** | | <number>  3 | <Name of Use Case>  <bring information> | |
| **Description of Goal in Context** | | <For showing information to users, system must have data.> | | |
| **Preconditions** | | <what we expect is already the state of the system>  -Users click “INFORMATION” button.  -Every requests from 2 are completed. | | |
| **Post Conditions, Success End Condition** | | <the state of the system upon successful completion>  System show correct information successfully. | | |
| **DESCRIPTION** | | < The use case description is a paragraph identifying behaviour, it comes from the requirements gathering>  Like said in use case 2, people want to find most of data in easy way. To provide them in fast, it should have data in itself. That’s why this information is in internal database. | | |
| **Main Flow** | | | | |
| **Step** | **Action** | | | **Alternate** |
| 1.1 |  | | |  |
| 1.2 |  | | |  |
| 1.3 |  | | |  |
| 1.4 |  | | |  |
|  | | | | |
| **EXCEPTIONS or ERROR Flow Description** | | | | |
| **Step** | **Branching Action**  < Exception number m of Use Case n> | | | **Alternate** |
| 1.2 | <> | | |  |
| 1.2.1 |  | | |  |
| 1.2.2 |  | | |  |
| 1.2.3 |  | | |  |
| 1.3 |  | | |  |
| 1.3.1 |  | | |  |
| 1.3.2 |  | | |  |
| 1.3.3 |  | | |  |
|  | | | | |
| **ALTERNATIVE or VARIATION Flow Description** | | | | |
| **Step** | **Branching Action** | | | **Alternate** |
| 1.2 | <> | | |  |
| 1.2.1 |  | | |  |
| 1.2.2 |  | | |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USE CASE** | | <number>  4 | <Name of Use Case>  <check flight information> | |
| **Description of Goal in Context** | | <After people put their information on system, system checks their information in double.> | | |
| **Preconditions** | | <what we expect is already the state of the system>  -Users put their flight information on system. | | |
| **Post Conditions, Success End Condition** | | <the state of the system upon successful completion>  Users are able to use specific place that could make group for travelling. | | |
| **DESCRIPTION** | | < The use case description is a paragraph identifying behaviour, it comes from the requirements gathering> | | |
| **Main Flow** | | | | |
| **Step** | **Action** | | | **Alternate** |
| 1.1 |  | | |  |
| 1.2 |  | | |  |
| 1.3 |  | | |  |
| 1.4 |  | | |  |
|  | | | | |
| **EXCEPTIONS or ERROR Flow Description** | | | | |
| **Step** | **Branching Action**  < Exception number m of Use Case n> | | | **Alternate** |
| 1.2 | <> | | |  |
| 1.2.1 |  | | |  |
| 1.2.2 |  | | |  |
| 1.2.3 |  | | |  |
| 1.3 | < > | | |  |
| 1.3.1 |  | | |  |
| 1.3.2 |  | | |  |
| 1.3.3 |  | | |  |
|  | | | | |
| **ALTERNATIVE or VARIATION Flow Description** | | | | |
| **Step** | **Branching Action** | | | **Alternate** |
| 1.2 | <> | | |  |
| 1.2.1 |  | | |  |
| 1.2.2 |  | | |  |

Non-functional requirements, management issues and decisions required to be made, can be identified in the following table

|  |  |  |
| --- | --- | --- |
| **RELATED INFORMATION** | Use Case: <1> | <Find flight or departure point> |
| **Priority:** | <how critical to your system/organization> | |
| **Performance** | <the amount of time this use case should take> | |
| **Frequency** | <how often it is expected to happen> | |
| **Channels to actors** | <e.g. interactive, static files, database, timeouts> | |
| **OPEN ISSUES** | <list of issues awaiting decision affecting this use case> | |
| **Due Date** | <date or release needed> | |

**Requirements Specification**

1. From the requirements analysis identified using the use Case scenario analysis identify key functional requirements.

Fill up the following matrix with *functional* requirements.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Req ID** | **Name of Req** | **Description** | **Priority** | **User Contact** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**System Modeling**

1. Identify possible *actors* in this system?

Users(travelers)

Developers

1. List the possible *use-cases* in this system?

# Verification and Validation of Requirements

1. Test Case Planning.

Put one sample test case in this section.

The additional test cases are to be in a separate test case word document.

Identify the User Acceptance Testing requirements for the use cases identified in Iteration 2 of the Use Case Analysis

Use the test case template to create initial Use Acceptance Test plans that will permit users and developers to agree the system will have been developed as specified by the requirements

Consider the test plan as a user guide or user manual for non-technical novice users of the system

|  |
| --- |
| **Test Case Number:** |
| **Test Case Name:** |
| Related Use Case  Name:  Number: |
| **Purpose:** |
| **Procedure Steps:** (Guided by Main flow or other flows of Use case) |
| **Expected Results:** |

# Completing the Feasibility Study

1. Before you do the final submission of the feasibility report, review and update your previous submissions.
2. Modify the requirements Specification list to high level (Abstract) core system features

**Update Use-Case Model and Requirement Specification**

1. Consider the Use case Model to be sure key functionality has been addressed in the analysis and modelling process.
   1. Do any of your use-cases need to be broken down further i.e., is there is too much functionality in one use-case?
   2. Update the Use Case Model as necessary.
   3. Update your RS with abstracted requirements as necessary.
   4. In your RS, put in a new column which shows what use-cases are related to which requirement.

**Prototype**

1. Sketch the home page/starting page of the app. You should take a photo of it and insert the photo into the document.

**Additional Functional & Non-Functional Test-Cases**

1. Write three additional test-case (using the test-case template) for each of three abstracted **high priority** *functional* requirements (one test-case per requirement/use case).
2. Write two test-case (using the test-case template) for each of the two **most important** *non-functional* requirements (one test-case per requirement).