Exercise 2

TICKET SHARE

Submitted by TEAM 52

Muhammad Fahad Rana
Nada Chatti
Maximilian Henneberg
Yasar Fatih Enes Yalcin

UML DIAGRAMS

User Related Classes

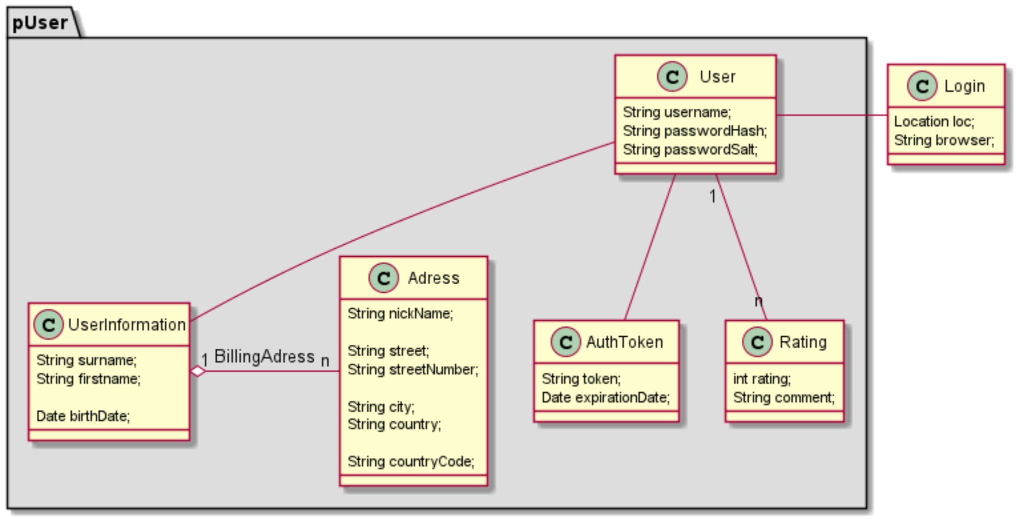


Figure 1. User related classes/models

The **User** model contains information about the user like name, password etc with 1 to 1 mapping with the respective classes. There is a mapping to **Address** which is a 1-N relationship for billing address used for escrow payments.

Along with the information, there is also **ratings model**(1 to many relationship) for it. This contains ratings which is left by others who joined their group. The average of the ratings(1-5) is the current rating of the user. This helps new users in deciding whether to join the group by the person or not.

An **auth token** is a 1 to many relationship which keeps track of the tokens and their expiry. The token is used by the rest api to handle authentication.

Groups, Ticket, Event and Participants

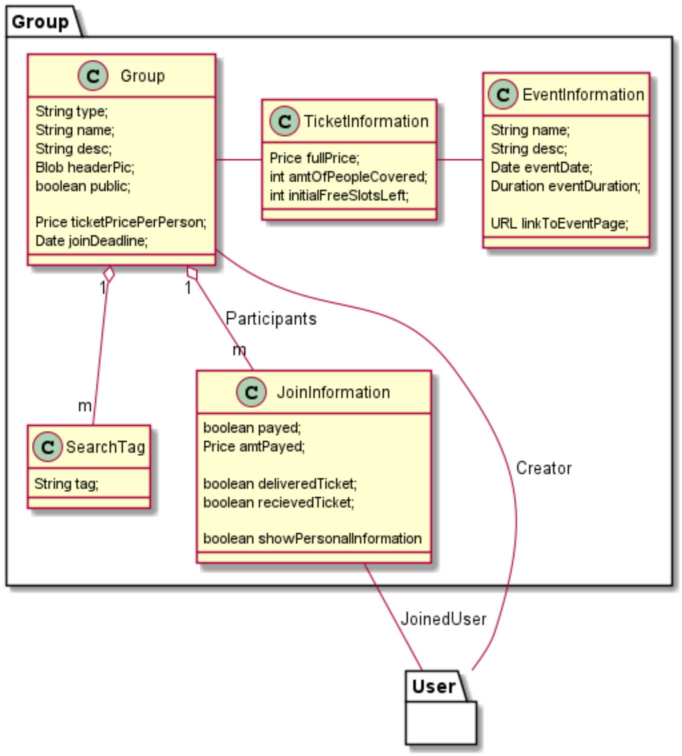


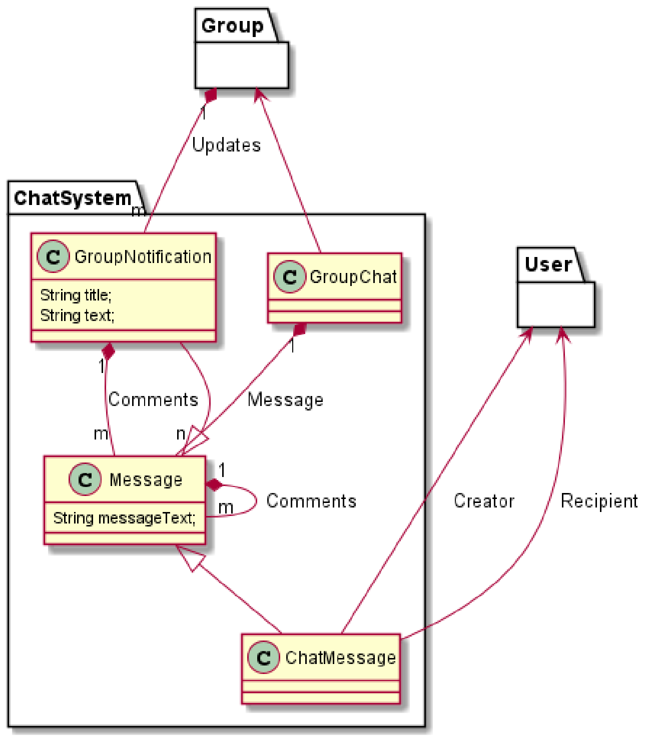
Figure 2 Groups, ticket info, participants etc

This UML has the backbone structure of our app. The **Group** is the ticket share model made by users offering group tickets. This model also contains the price per person.
A 1-1 Mapping with **TicketInformation** Model contains the information of the original ticket and same with Event Information for giving healthy information to people joining the group.

The **Join information model** is a 1-M mapping with the **Group** because 1 group can have multiple participants. This Model is also connected to **User** model to keep track of the users registered for the Group. The data elements deliveredTicket and payed gives the application information about the joinee.

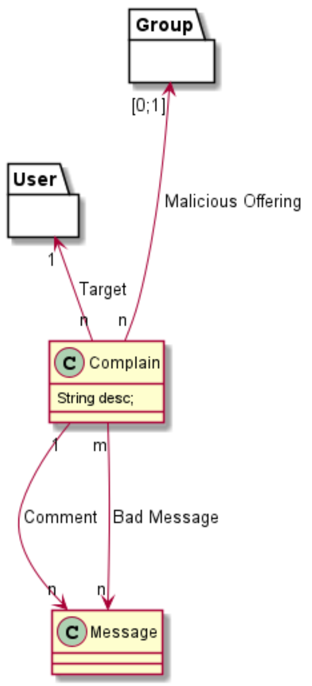
The **SearchTag** is a 1-M with **Group** which contains tags for efficient search results.

Chat system



This ChatSystem contains model related to chat functionality.
The **ChatMessage model** is connected to **User** which sends the message also another relationship which tracks the creator. Then there is **Group Chat model** which connects the chats/ messages to the group. The **message model** is connected to itself with comments which gives us information of replies/ comments left for specific messages.

Support



This a comparatively simpler UML. This contains classes which will help us get complaints from the Users. The **User** files a **Complain**. This then contain's either a abuse **message** in a chat or malicious offering. For example a user sees a fraud offering tickets. They can report us the **group** and we can then look into it.

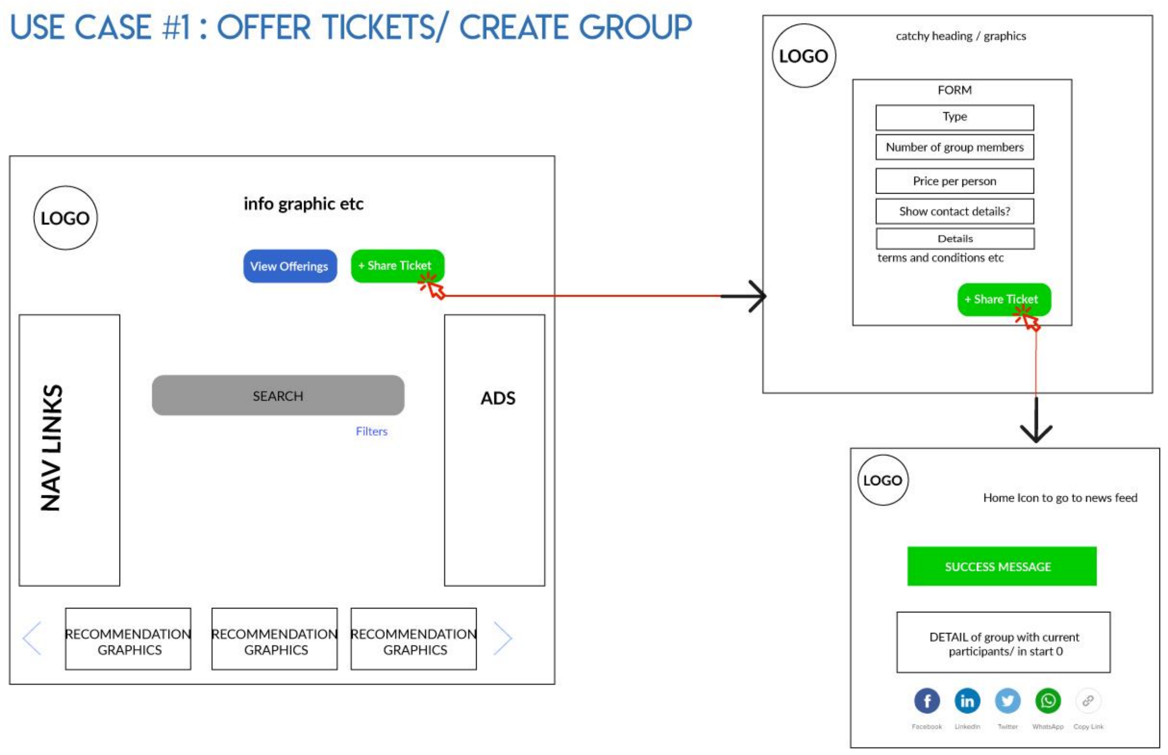
There is a 1-n Relationship of User to Complains because one user can file multiple complains. Then there is a m-n relationship of **complain to message model** because multiple complaints can be about an abuse message or comment. There is 1-n of **Group with Complains** because for a group there can be multiple complains.

Use Cases

Use Case 1: Create Group for ticketshare

The user who wants to create a group can simply sign-in and on the homepage there are buttons which are easy to find. Share Ticket button takes the user to another page where he/she can enter the details related to ticket(like price per person, event etc) and then post it. Once successful, the user will see success message along with shareable links to different social media profiles.

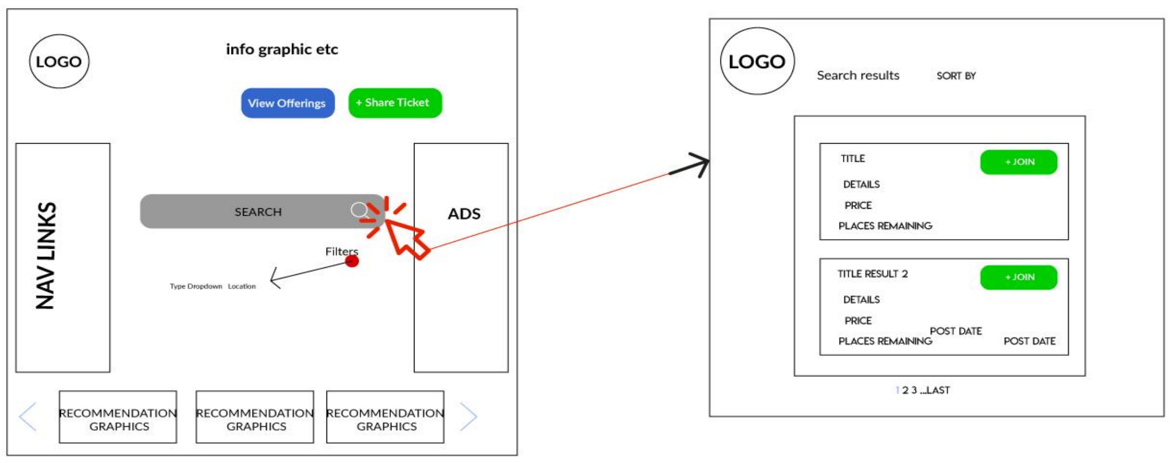
Below is the use case diagram of this use case which depicts the workflow:

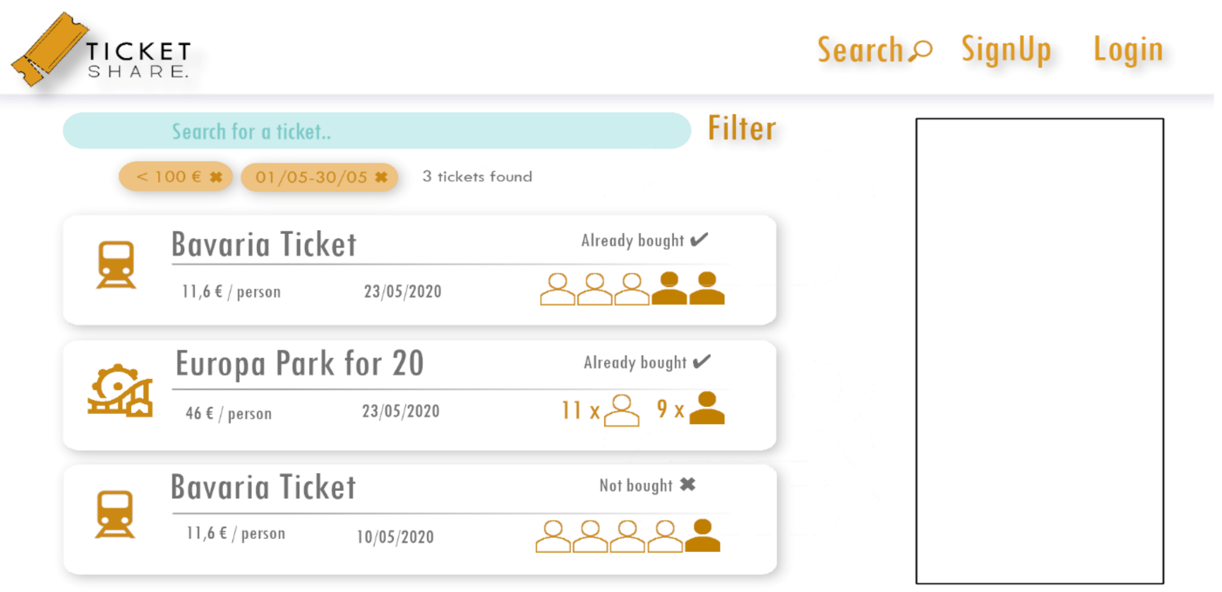


Use Case 2: Search for a group

After authenticating, the people will see some results related to them on their homepage in the form of recommendations and maybe a couple under the search bar(depends on whether it jeopardizes the performance in terms of Page Load Times etc)
Also, the search bar and filters will show them a list of search results based on their search criteria. The search tags we stored in the model along with the title and description would be used to carry out the search algorithm. The user can also sort the results which are shown in a comprehensive way.

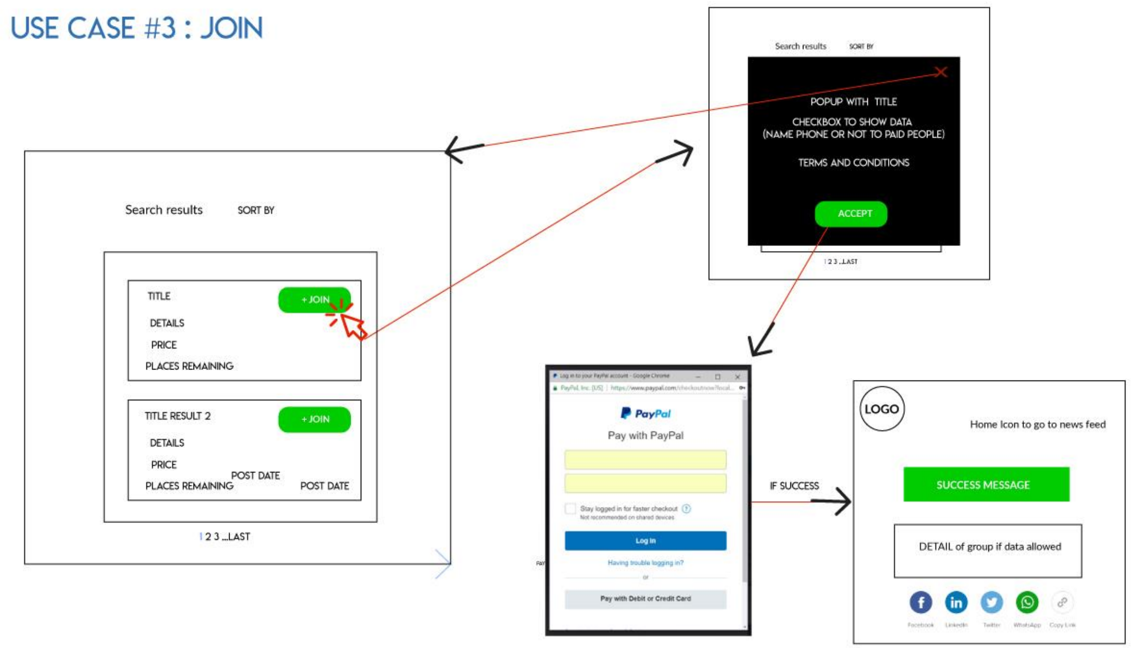
USE CASE #2 : SEARCH





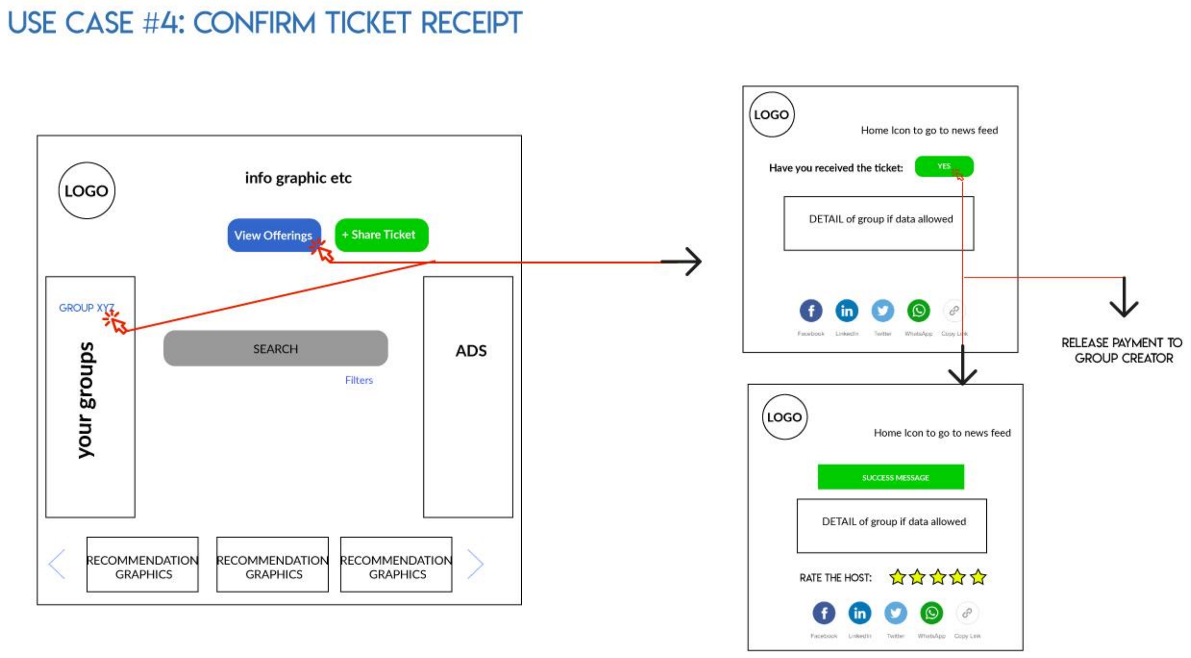
Use Case 3: Join a Group

This use case is a continuity of the previous one but it is a use case in itself. After getting the search results, the users can simply click join. A popup will show up with terms and conditions and payment info. We will try to keep the text in this popup minimal. Once clicking on confirm, the paypal payment window will open up which will use Paypal Order API. The user will pay their share and we will have the money in our escrow. We will show them the success message along with the shareable links(for more reach).



Use Case 4: Confirm Ticket receipt

Once user receives the ticket by the group organizer(they will use our chat to carry out the communication), they can confirm ticket receipt and rate the user. This will trigger the payment release event and will send the payment to group organizer.


USE CASE #4: CONFIRM TICKET RECEIPT
LOGO
info graphic etc
View Offerings
+ Share Ticket
GROUP XYZ
your groups
SEARCH
Filters
ADS
RECOMMENDATION GRAPHICS
RECOMMENDATION GRAPHICS
RECOMMENDATION GRAPHICS
LOGO
Home Icon to go to news feed
Have you received the ticket:
YES
DETAIL of group if data allowed
Facebook
LinkedIn
Twitter
WhatsApp
Copy Link
RELEASE PAYMENT TO GROUP CREATOR
LOGO
Home Icon to go to news feed
SUCCESS MESSAGE
DETAIL of group if data allowed
RATE THE HOST:
Facebook
LinkedIn
Twitter
WhatsApp
Copy Link