CS 319 - Object-Oriented Software Engineering

Final Report

Your Story

An Online Messaging Portal for Text Based RPG Games

Group 22

Ali GÜNEŞ

Cevat Bariş YILMAZ

Erin AVLLAZAGAJ

Ertuğrul AKAY

Course Instructor: Uğur DOĞRUSÖZ

Contents

Introduction

“Your Story” is an online portal for text based role playing games. The game itself is completely aimed to help users hosting an RPG Lobby or joining a Lobby depending on their interest. It has completely different gameplay from the other games, since Your Story integrates a messaging application with a voting system, fresh new characters and open ended story lores to develop on it. It is also designed for desktop computers, since writing on a keyboard is the easiest and fun way to writing for these type of gamers. The purpose of the game is having fun by chatting with the other players in order to develop a story and acting like a character you always wanted to be. The idea behind this project is to bring together a group of story writers to develop a story their way and get to know other interesting people in the world. This game’s purpose is to help the players develop their imagination and collaborate to create a great story or even connect with other people sharing the same talent and interest.

This game is different from many similar game because the user is more valued to his writing and creating skills rather than just valued by how fast they click the mouse. Another difference to other games is that the ending of the game is not decided on when the time ends or the “princess is rescued”, the ending is let free for the player to decide democratically. The same way is for starting the game or kicking out toxic players.

In the development process, we are focused on colorful yet simple palettes and patterns to create a unique way of showing lists and profile. With its small size, it also does help to do multitasking, like when you are writing to make your character speak you can also use Youtube or do any other stuff you want. With using a database and login system, everything about your games and profile is safe and secured in our database. Iin case of a system shutdown or internet connection disturbance, you can continue to play with your friends from where you stopped.

In summary, Your Story is focused on a specific type of gamers, and developed in order to make socializing and chatting easier for them. When the voting system prevents unnecessary disturbance, story lore and characters give them a kickstart for developing a story by themselves.

Changes to the design

The changes in our design were not that huge. The UI created some more classes for the listeners and some private methods for code reuse which were scheduled to run at specific times. The private functions were designed to be called frequently so where as efficient as possible. We also didn't consider the scheduled caller objects in our design so in the implementation those were also part of it. Also we forgot some minor view components in the view classes like a text field in LobbyCreatorView. We also forgot a few functions in database communication but overall there were no major changes from the design. The layering remained the same. Also the architecture of database layer stayed same but we just had to add some extra methods to get some data we didn't think about before. Mainly, the middle layer had some improvements to make the player-system interactions more safe and fast and thus most of the methods that are used in the middle layer classes are integrated with the server.

UI layer

The UI hasn't changed much in appearance regarding what we described in the final design report. The main functions exist for all. We added a lot of private functions because we didn't consider the synchronization functions that were to be called from time to time. Also a major change in the design was the change in interface. The functions have the word "View" added to the end. This happened because when overriding the show() function it was overriding another deprecated function of the swing classes and so we decided changing the name rather. This didn't happen in the case of character class which stayed the same and we just removed ambiguity by importing explicitly importing it. Other changes are seen to be the new classes we created that serve as listeners, like ProfilePicMouseListener, WaitingLobbiesMouseListener, FinishedGamesListener, etc. Also, we added some subview classes for making the project more understandable. Other classes we created are the Runnable classes that we needed in order to make a scheduled run of the private functions are mentioned above. Some of them are for example: ProfilePicUpdater, OnlinePlayersUpdater, OngoingUpdater, LobbyUpdater, etc.

Middle layer

The middle layer, which includes the main logic behind the game and interactions between database side has kept the general logic same as in the design process. But, there are some extra methods added, and some old methods are changed in a way that their parameters are made to be compatible with server side.

Since the implementation process is more complex and detailed due to unknown system and database errors, most of the methods in java classes are fully dependent to the third layer, database side. The reason for this is to prevent user-system interaction collisions and keep users' information safe. This means, when a method creates a new object by the request of a user, firstly the database checks if this action is applicable and does not collide with other users' action and then allows method to execute itself. For example, by this way we can block two users entering a lobby which has only one empty seats left. Thus, only the first user that clicked the enter button can enter the lobby, and since the quota for the empty seats are updated in the server side, the second user is not allowed to enter.

Also, another significant change that we did is in the methods that return objects and take objects as parameters, we changed them to take object ID's which are stored in the database. This way, we do not change objects directly, but with using their ID's the database change them for us.

Database layer

We were able to protect main shape of the layer. However we had to add extra functions to use them in extra parts we didn't consider at design step. Like setting the state of a lobby or getting the character informations of a story. Also we had to added some private functions to convert the result set we got from database to objects. Also since it was easier to handle the data coming from database as array lists in other layer, we changed the type of multiple results to lists from arrays. Although the return type of DBInterface stayed as arrays, the change was made in other classes that directly used by middle layer. Finally we had to added extra static table names to database classes for new implemented functions.

Exposed Documentation

The Java API Doc shows the exposed documentation of Your Story, in HTML form. The file is uploaded in github.

User's guide

System Requirements

The requirements for this program are quite simple. The machine should have Java 8 SDK installed and configured properly and configure it to be the default program that opens .jar files. The system should be able to satisfy the requirements of Java 8 (like the graphics, memory, etc) and have a decent internet connection over DSL. The Dial-Up connection might be too slow for the data being transmitted.

How to build

The master folder in the github has a build.xml file which takes care of all the building. By running "ant test" command in the master directory the ANT will evaluate the build.xml file and return successful. For that the README.md file contains the build status from https://travis-ci.org/Albocoder/CS319-Group22 which already has passed and is working. Running "ant build" will build a jar file out of the project and then by assigning execute permission to the newly created file a simple double-click will load the file in JVM and the game will be up and running. The user will be able to view the login screen and prompted to enter username and password.

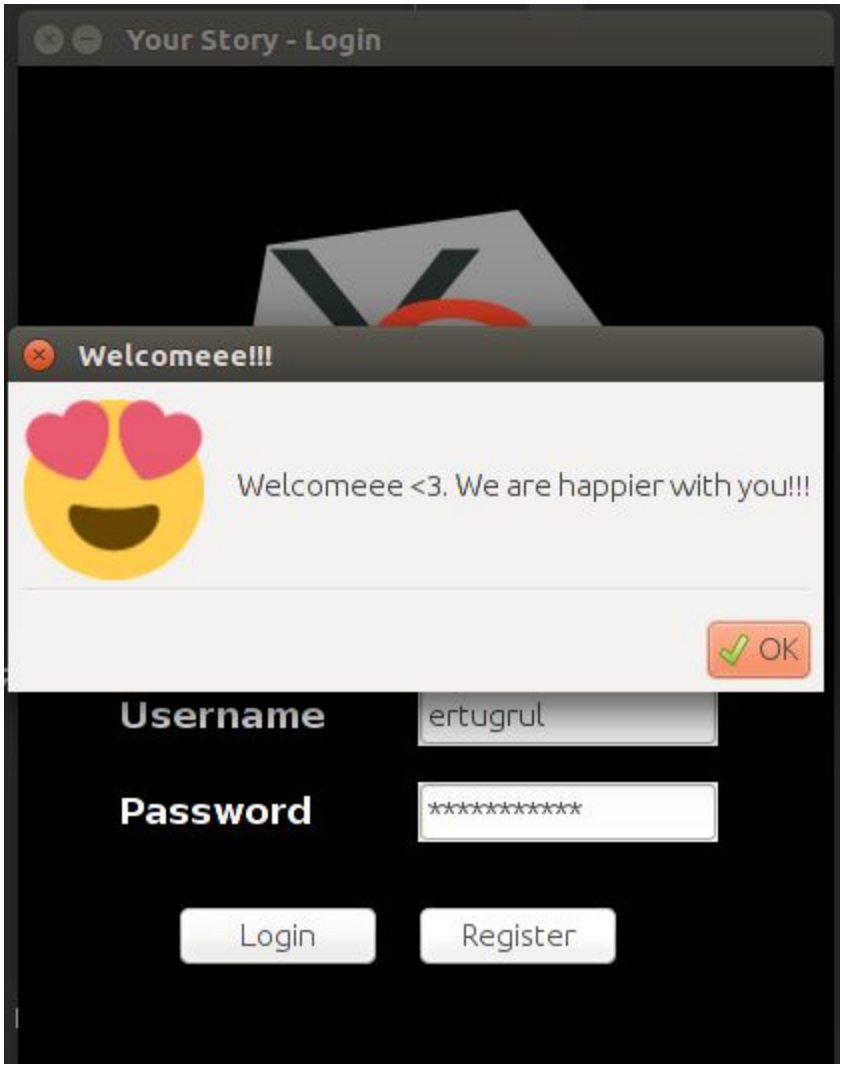
How to install

Installation is pretty easy. The game consists of only the jar executable file. No installation is required. It is portable and can be copied to any machine. Once it exists it is installed.

How to play

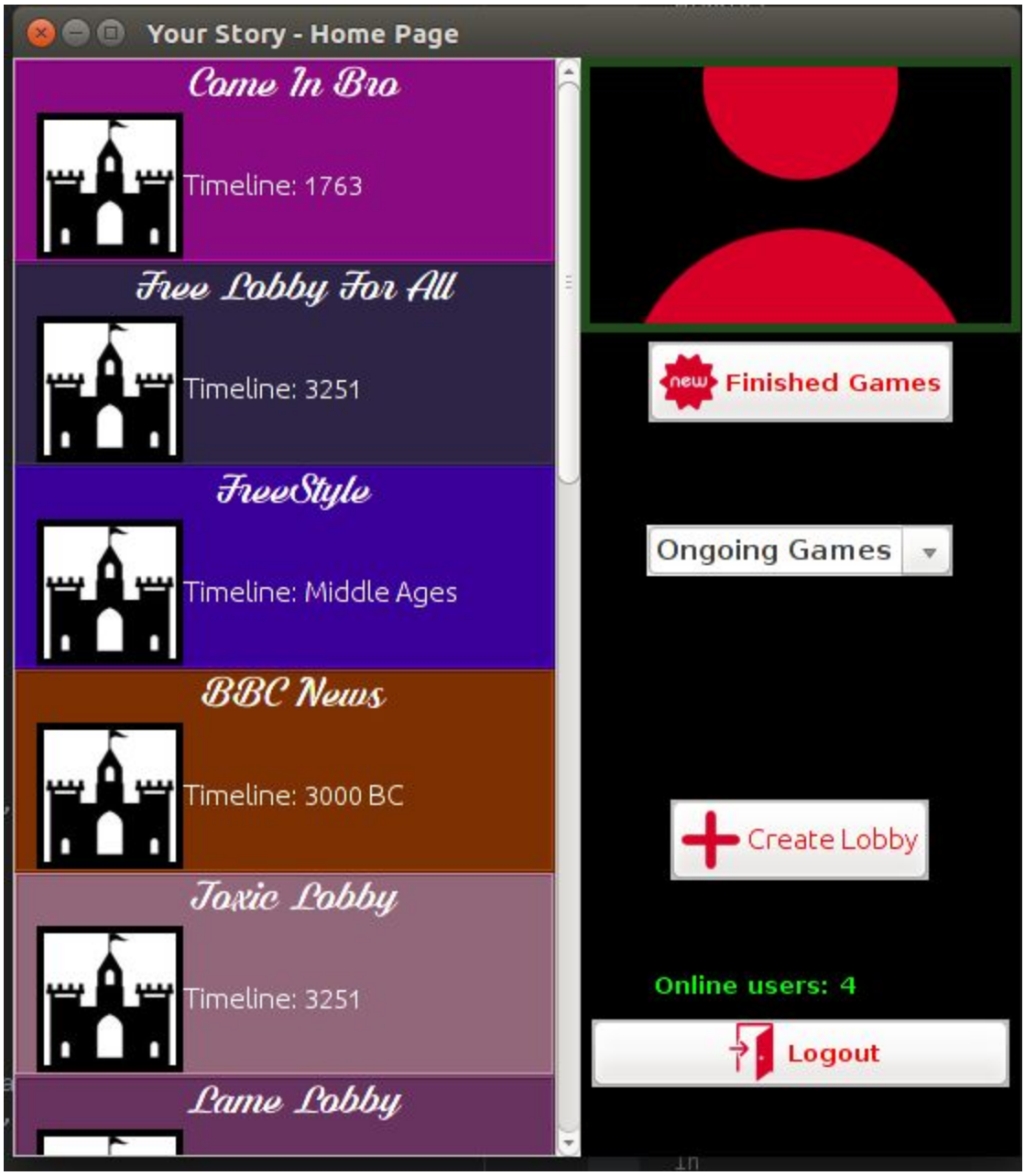
The game will start at the login screen. Form then the user is prompted to enter their username and password if the player already has one. If not we have made it very very easy for everyone to join.

All you need is to fill up username and a password field with you desired credentials and press register. If the username was not taken by anyone the player will be acknowledged with a heartwarming (literally) dialog box and be ready to view their homepage.



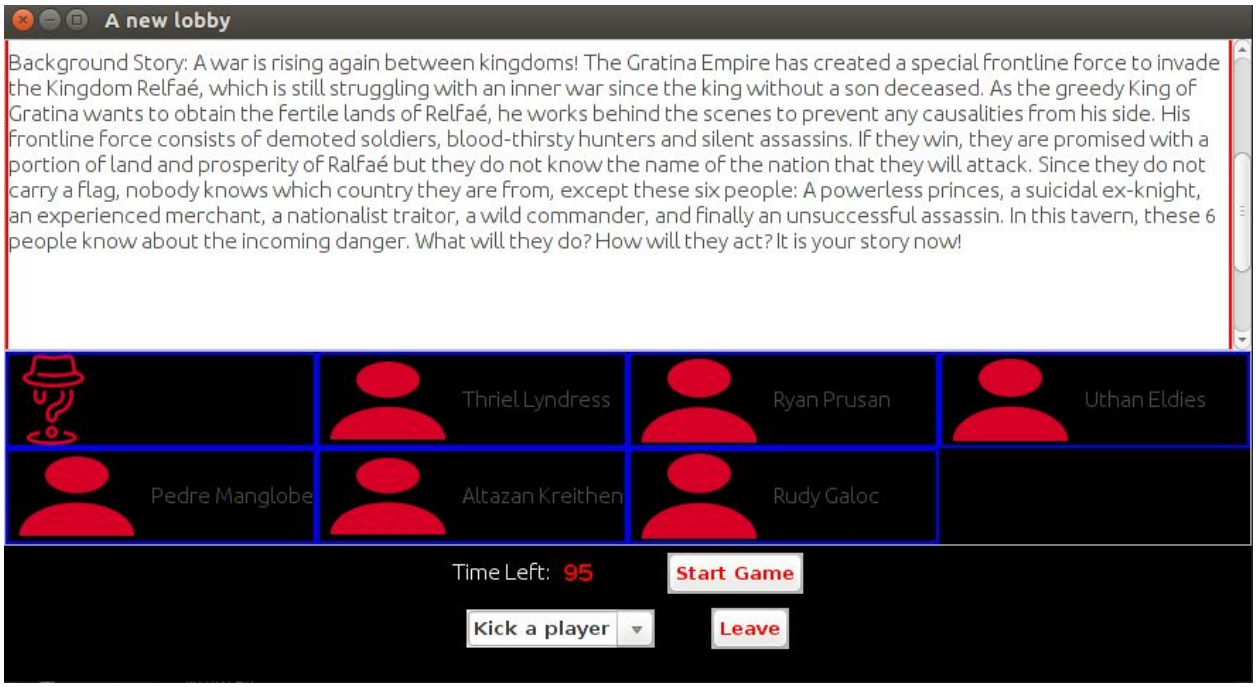
From there the player can change their profile data but, at the first release, not the profile picture.

The lobbies currently waiting for players are showed in a list at the homeview screen. Player can either choose a waiting lobby to join, or create his own lobby.

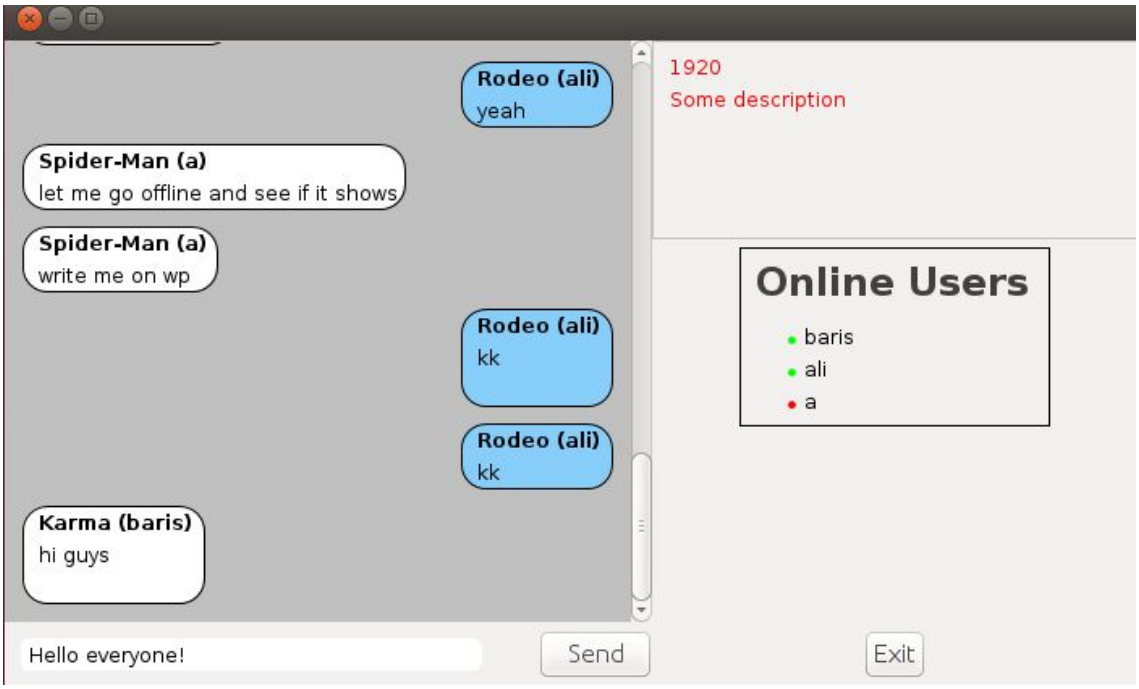


If the player choose to create a new lobby, he will be prompted to a new page to select a story, and the name for the lobby. In the list of stories, every story has a description, timeline and list of characters. Player can choose the story he is interested, and click on it. If the player entered another lobby, this screen would have not shown to him.

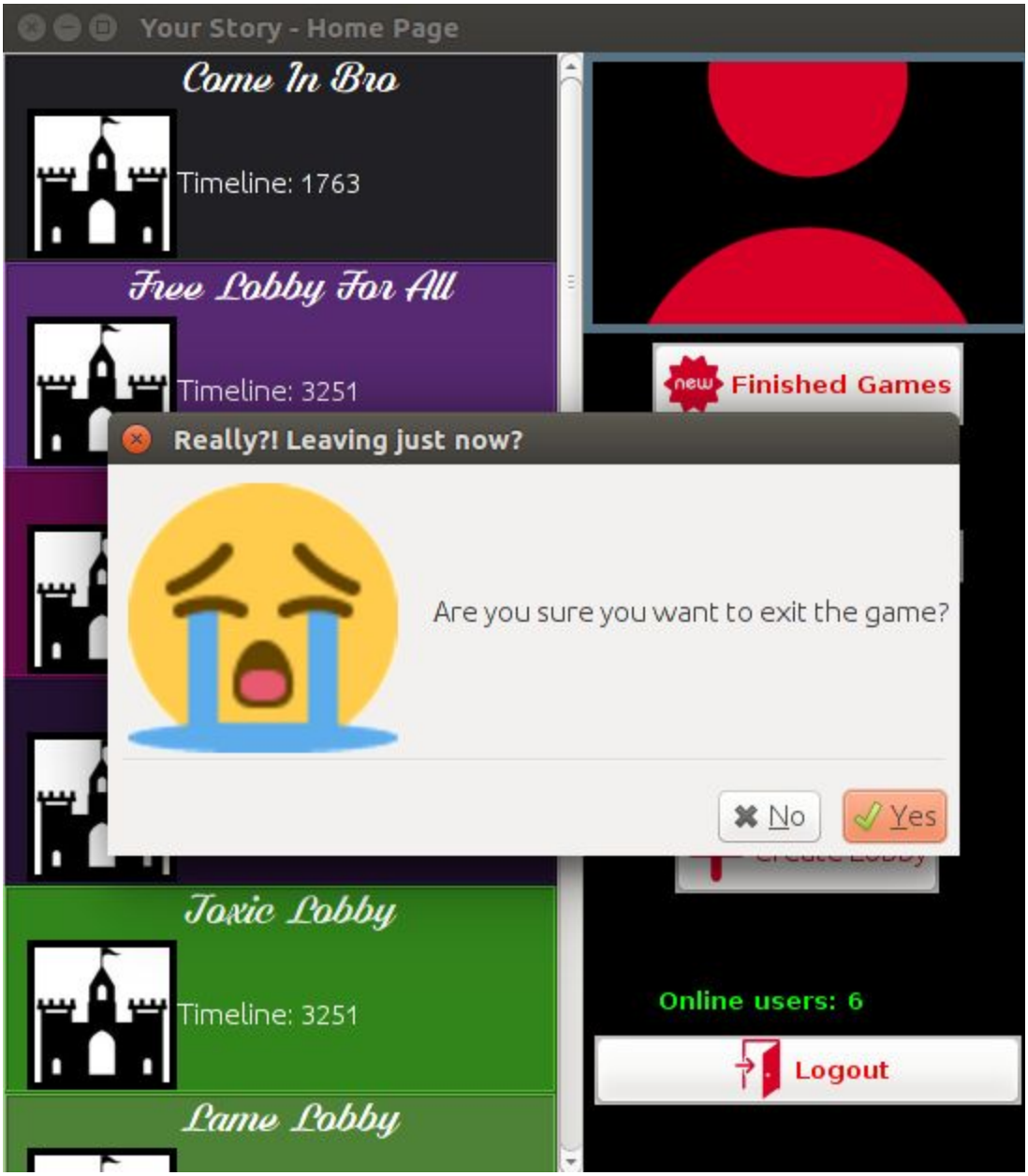
Then, a new screen will appear for player to choose a character. In this screen, player should select a character that he wants to play in the remaining time.



After choosing a character, game will start at last. The description will still be on the screen, so the players can remember as which character they are acting and develop the story as messaging each other.



Players that are inside this lobby, can continue their games anytime they wanted. Their messages will be stored in the database, so after they exit and log in again they can continue the game from where they stopped.



What's left

Since the project was not fully finished there are a few small features that will be added in the next release.

1. The image support for the profile, seat, character.
2. Another feature we didn't have time to implement was the voting which has the same mechanics as the chat system, if a new voting is called a new row is created in the database and the clients (which update every time with the voting table just like with the chat) get the new row of the table and display a dialog that prompts the user to enter vote then the vote is sent and after some time the result is called by the client which changes the state of the lobby immediately depending on the answer. Everything is already set up in the database and its classes, however the middle layer didn't support that feature yet so the UI can't
3. The Viewing of the finished games is not supported.

End User License Agreement

Our software is protected by the agreement signed in December/24/2016 by the team leader Erin Avllazagaj. For more legal concerns please feel free to check the website below
https://www.binpress.com/license/view/l/819def9e2fbc3efcb40e918a0a66f27d
The license is included in the bundle of the release and in github. Our software doesn't violate any 3rd party or author's copyright and is crediting the authors of sources like images used. The licenses like the one of Freepik is legally allowing us to commercialise our product by crediting them.