Adilene Alaniz
David Davis

Night Board! - Final Report

Description

Night Board! Is a family board game to enjoy playing with your friends or family to have a fun but different and dynamic night. Night Board! Consists of a game(s) that contains a Tic Tac Toe virtual “board game(s)” in which two users can play against one another from either one or two separate electronic devices. The users have the option of restarting the game and challenging one another to multiple wins. The winner of Night Board! Is the one who earns the most points. The users will also have the opportunity to share their hard-earned scores via social media(Facebook, Instagram or Twitter). The app will also include sound/music to make the games more enjoyable however, they will also be provided with a silent option for quiet gaming.

Design/Implementation

After a lot of time and research we decided the game implementation was not as much important as the added on features. With assistance we were able to produce a simple game of Tic Tac Toe. The first feature is we decided to add sound and/or special effect sounds when playing the app. Given that not everyone is a fan we also added the option to silence these sounds. The second feature that was implemented was to add background music to the game where *audioplayers* and *audio_cache* APIs to manage the play and pause states of the mp3 file. was Another feature we implemented was social share. Social share is a plugin which will allow any user to share their scoreboard via social network. The version we used allowed social share through either Instagram, Facebook or Twitter. This plugin required the applications API’s to be installed prior to being able to successfully login and share. One of our most interesting features is we implemented.

Challenges

Our original approach was to develop a truth or dare game. However, we soon realized the implementation was a lot like our Quiz App assignment and we wanted to try something new. Because we changed our mind so many times we came to realize that timing was out biggest challenge as we were losing time. We then decided to turn main focus on our bigger and more important features. One challenge was implementing the ability to share on social media. To begin, we had to decide how to implement and which plugin or API to use. Facebook offered their own plugin with a matching company icon which was a neat but concise approach. The issue was not all others offered the same features. So our best and most convenient plan of action was to go with the social_media_plugin. This plugin however came with its own set of challenges as we had to create developers Facebook profile and obtain an Application key for this and any other "testing" applications ran as well as a client token. That was unfortunately not the end as this plug in also required permissions and authorities, our Manifest XML got filled.

Another challenge we encountered was connecting making two machines running the same app and playing the game in real time. throughout the research we made, we found various approaches and APIs to use to accomplish connection between two machines where the two most easier approaches were web sockets or Bluetooth sockets. The challenging part was to find a way to make them run in real time, We could not find any source of tutorial or information of how to make this work.

Technical Lessons

Flutter overall is difficult to fully understand as there are so many options and varieties to choose from. More so with having to create your own project. The most important lesson we would both agree on is that, because of that variety, research is the key. It did help to have our class diagram prewritten so we had an idea of what to search and what to expect. We spent a good portion of our project time planning and researching widgets and looking up information that could help us in creating our own version. We also spent time viewing flutter tutorials which aided in creating our implementation of the game.

We also agree that making an app requires to not only code but to understand the basics of APIs that you plan to use because it would help you when you are deciding which of all the APIs to use for desired task. We encounter this when implementing the web socket connections when we just wanted to learn as we were coding but then we stopped for a moment and researched over how socket connections work in general and in apps without a server, leading us to a better understanding on how to implement this feature efficiently.

Project Complexity

672 lines of code (so far)
7 classes (so far)

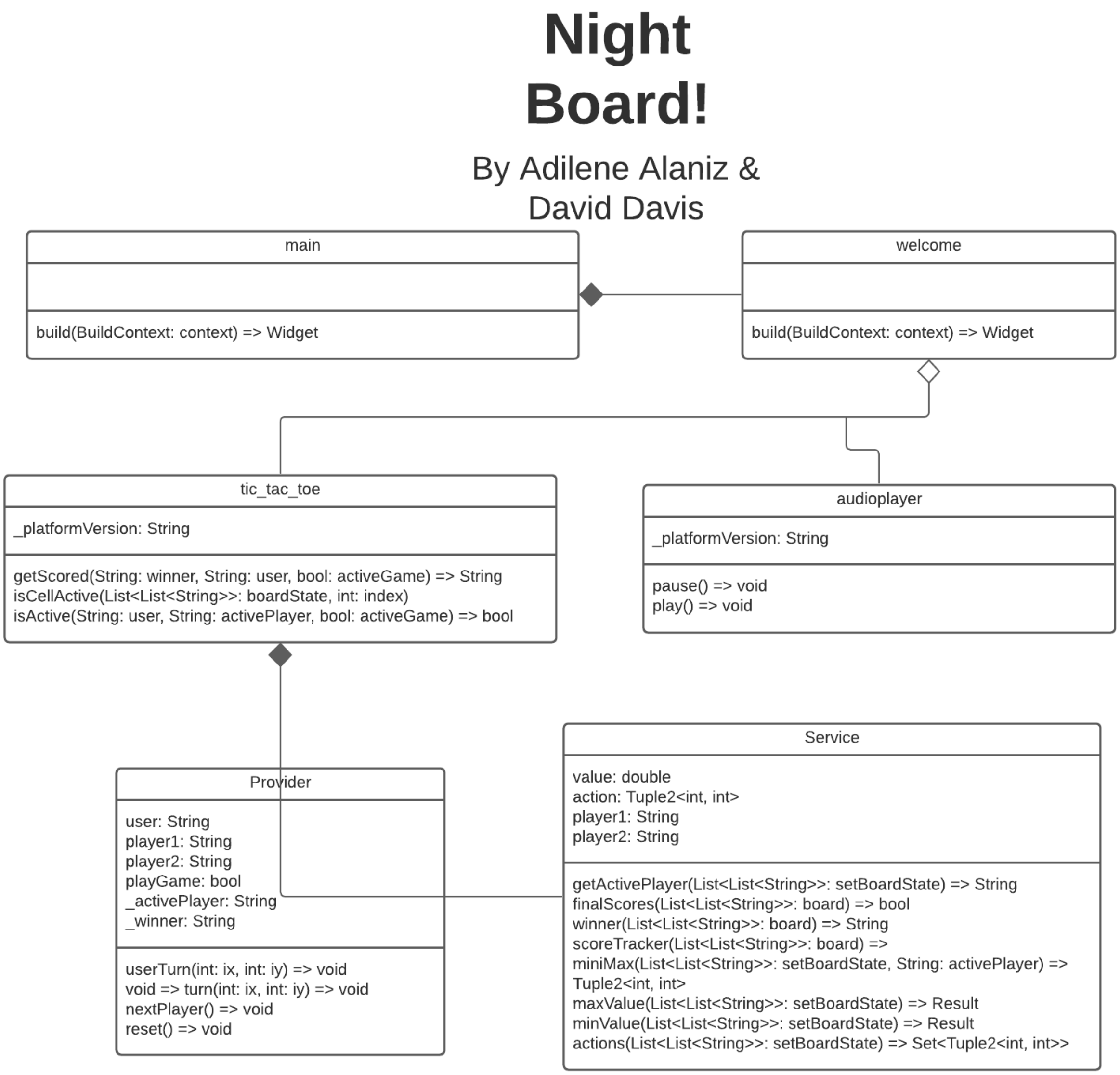
Libraries:

social_share_plugin - for live social media sharing
permission_handler - grant permissions to save files on our devices as well as share
flutter_toast - library that supports error/validation messages, used for image saving/uploading
image_picker - allows user to select image to share or upload
screenshot - plugin allows easy access to screenshot on either widgets or full screens
audioplayers – allows to manage audio files
audio_cache – allows to store audio files and access them as cache materials

Features:

We created our Appcolors() class to be able to customize our colors used throughout the app

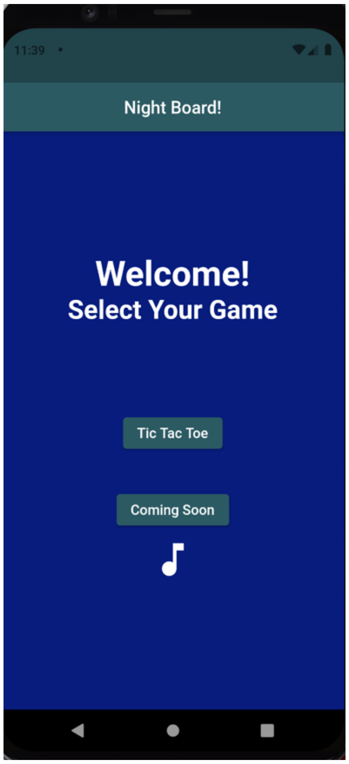
Class diagram


Night
Board!
By Adilene Alaniz &
David Davis
main
build(BuildContext: context) => Widget
welcome
build(BuildContext: context) => Widget
tic_tac_toe
_platformVersion: String
getScored(String: winner, String: user, bool: activeGame) => String
isCellActive(List<List<String>>: boardState, int: index)
isActive(String: user, String: activePlayer, bool: activeGame) => bool
audioplayer
_platformVersion: String
pause() => void
play() => void
Provider
user: String
player1: String
player2: String
playGame: bool
_activePlayer: String
_winner: String
userTurn(int: ix, int: iy) => void
void => turn(int: ix, int: iy) => void
nextPlayer() => void
reset() => void
Service
value: double
action: Tuple2<int, int>
player1: String
player2: String
getActivePlayer(List<List<String>>: setBoardState) => String
finalScores(List<List<String>>: board) => bool
winner(List<List<String>>: board) => String
scoreTracker(List<List<String>>: board) =>
miniMax(List<List<String>>: setBoardState, String: activePlayer) =>
Tuple2<int, int>
maxValue(List<List<String>>: setBoardState) => Result
minValue(List<List<String>>: setBoardState) => Result
actions(List<List<String>>: setBoardState) => Set<Tuple2<int, int>>

*****See Appendix bellow**

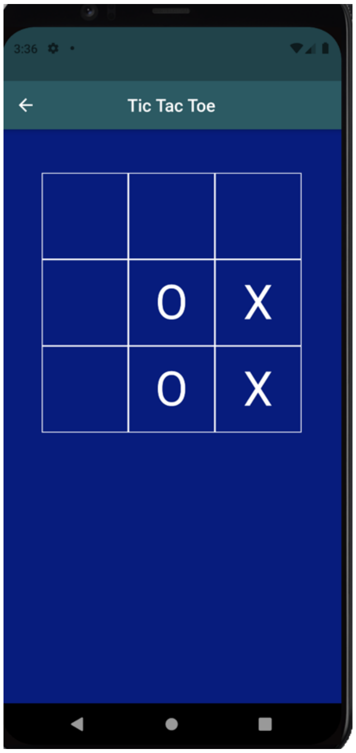
Appendix

Welcome page



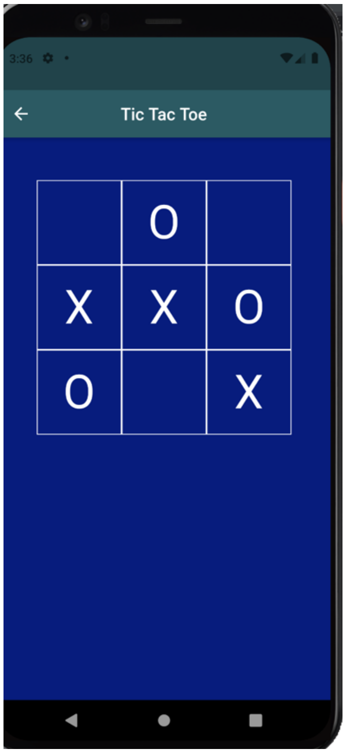
Welcome!
Select Your Game

Game sample run



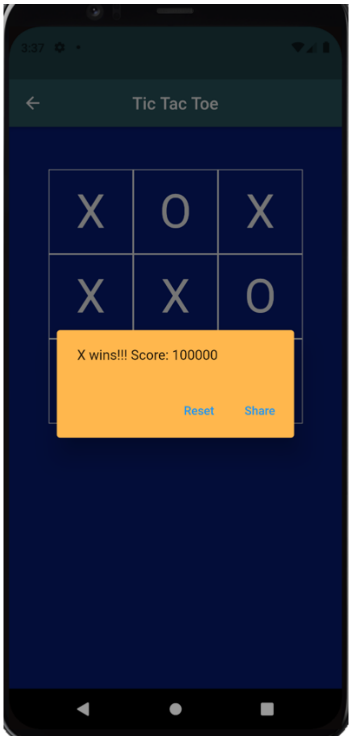
	O	X
	O	X

Game sample run



	O	
X	X	O
O		X

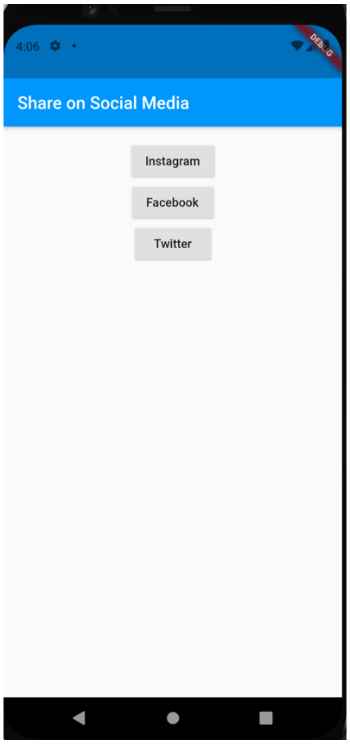
Sample Score Board



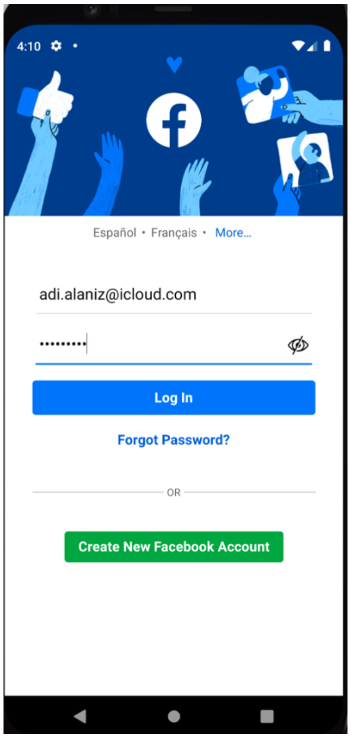
X	O	X
X	X	O

X wins!!! Score: 100000

social_share_plugin



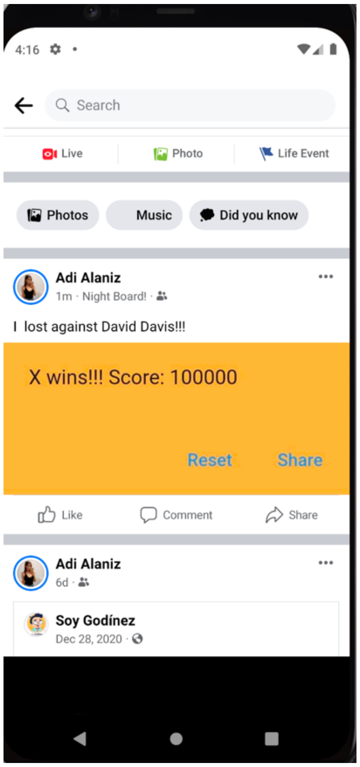
Facebook login



adi.alaniz@icloud.com

•••••••••

Facebook feed



Adi Alaniz
1m · Night Board!

I lost against David Davis!!!

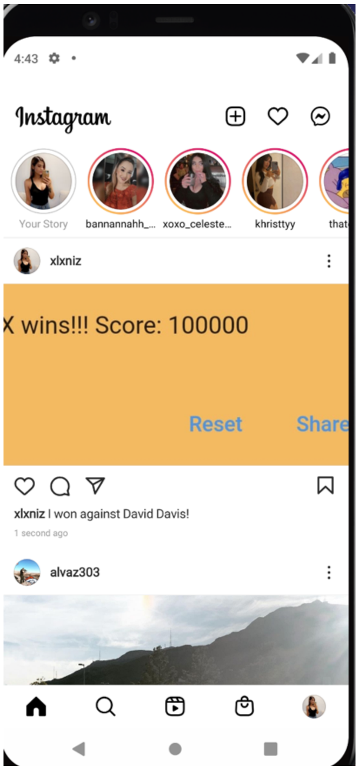
X wins!!! Score: 100000

Reset Share

Adi Alaniz
6d

Soy Godínez
Dec 28, 2020

Instagram feed



xlxniz

X wins!!! Score: 100000

Reset Share

xlxniz I won against David Davis!

1 second ago

alvaz303