Multiplayer Pong Game Program Narrative

Team Members:

- Nael Louis (1934115)
- Rodrigo Rivas Alfaro (1910674)
- Daniel Lam (1932789)

Roles:

In general, every team member has worked equally well on their side of the project. Even when not working on a certain aspect of a project, each team member has contributed to the project by proving help and advice on what to do.

Nael:

- Redesigned and worked on Multiplayer and finalized it
- Sync up paddles, player scores from client with server
- Custom menus (Main Menu & Game Menu)
- Save and load functionality of Pong
- JavaFx components to set up Bat and Ball components
- Add login and password dialogs
- Cleaned up code

Rodrigo:

- Contributed to start of Multiplayer
- Creation of keystore class
- Password hashing and verification using SHA3-256 algorithm
- Symmetric-key encryption/decryption of save file
- Signing and verification of signature of PongApp file
- Cleaned up code

Daniel:

- Contributed to start of Multiplayer (sync walls and ball)
- Added error dialog boxes in case of invalid IP addresses or 2 hosts on same IP address
- Sync up pausing on Game Menu between client and server
- Input Validation on IP address
- Add Documentation (program narrative, README, etc) & OWASP dependency check
- Add functionality for client saving and loading by pinging server

Overall, we are all very satisfied with the project and the effort put by every team member!

Description:

This project consists of a Multiplayer Pong Game originally created by Almas Baimagambetov. Pong is a 2D sports game in which two players try to score against each other while controlling paddles up and down until one player reaches the maximum of 10 points. In our version, the game has been modified and altered so that it implements a Multiplayer functionality for two players to connect with each other through IP addresses and play simultaneously (one of which is host, the other is client). Players also have the option to save and load the state of their game and their files will be safely encrypted and decrypted using a SHA3-256 algorithm.

The project is based on Java and a Java Game Library called FXGL which provides the functionality for the server and client. GUI is implemented using the JavaFX library and GUI elements from FXGL.

How To Build:

The project contains many necessary dependancies that are crucial for the project. Project can be run on Windows and Linux machines.

1. Set up Java on your machine by installing the latest Java JDK and SDK.
2. Clone the project and open up NetBeans.
3. Verify that all the dependancies have been installed (FXGL Library).
4. Build the project and Run it.

If everything goes well, you should be led to the Pong Main Menu screen!

How To Play:

OBJECTIVE: Score 10 points before your opponent by sending the ball to the other side!

1 player only:

1. Launch the application twice to have two Pong Game window and select "New Game".
2. Select one window as the host first and then select the other as client.
3. On the client, connect to your current IP address (use ipconfig on cmd) or connect using 127.0.0.1 (localhost).
4. Once you've automatically connected to the game, play and win!

2 players:

1. Launch the game and select new game.
2. Have one user selected as host and the other as client.
3. On client, type in the host's IP address to connect to the server (localhost won't work here!)

4. Play and win against your opponent!

Information can also be found on the README.md of the project's Gitlab repository.

History:

Project & Multiplayer History:

- Initialize project and implement base PongGame from AlmasB
- Set up MultiplayerService.class inside PongApp settings
- Add onUpdate(), modified initInput(), onServer() and onClient()
- Modify and adjust Bat and Ball components and Connection<Bundle> in PongFactory.java
- Add error checking dialog boxes **(see *IP Address & Dialog Boxes*)**
- Sync up Pong game on both server and client side so that paddles and scores automatically update on each side
- Add save and load game functionality **(see *Save & Load*)**
- Add custom main menu and game menu **(see Menus)**
- Change color styles of application to fit theme.
- Sync up pausing on server and client side of the game
- Set up cryptography for Pong (see ***Cryptography (encryption, key pairs, SHA3-256 hashing, etc)***)
- Add Documentation (Program Narrative, README.md, comments, etc.) & OWASP dependency check
- Add functionality for client saving and loading by pinging server

IP Address & Dialog Boxes:

- Add dialog box for IP Address Input.
- Add error checking for invalid IP Address.
- Add error checking for valid server IP Address but no host
- Add error checking for two hosts sharing on same IP Address
- Display error dialog boxes to user
- Add Input Validation on inputs

Save & Load:

- Add onPreInit()
- Saving: Store player scores into Bundle
- Create save file out of saved data
- Load: Add loadLastGame() to show dialog box for user to reload save file
- Loads player scores and ball positions back into game

Menus:

- Remove default main menu and implement custom menu
- Create new PongSceneFactory() that returns MainMenu and GameMenu
- Create PongMenuButton to add display and functionality of each option in a Menu

- Create custom Main Menu and Game Menu using JavaFX

Cryptography (encryption, key pairs, SHA3-256 hashing, etc):

- Create KeyStoring.java class
- Add key generation methods (createStoredKeys(), loadKey(), saveSecretKey(), GenerateKey())
- Add password ProtectionParameter, SecretKeyEntry and getters for keys
- Store SecretKey into KeyStore
- Add tests for encryption
- Implement SHA3-256 algorithm (MessageDigest, bytesToHex())
- Add encryption and decryption for files using Cipher
- Add file signing and verification of file signing using Signature
- Add initialization vectors using GCMParameterSpec and IV
- Add user input password checking using SHA3-256 hashing
- Implement encryption to save & load features of game