Team 1: Scott Crisenbery, Derek Nasfell, Alex Newkirk, Cody Simpson

**Part 1: Non-Functional Requirements**

(Platform, system compatibility, overall quality requirements)

1. The system is to be a local Windows application
2. Deployment Environments
   1. Windows 7
   2. Windows 8
3. The system is to be installable
4. The system will be thoroughly tested prior to delivery
   1. No functional errors in 25% of delivered requirements

**Part 2: Functional Requirements**

**Glossary**

*Classic Mode:* The game starts at a simple slow pace and slowly speeds up the rate at which the blocks drop until the stacked block reaches the top of the play area, ending the game. Scoring is tracked. The game ends when topping out occurs.

*Timed Mode:* Timed Mode challenges the player to score as many points as possible within a two-minute time limit. The game ends when topping out occurs or the time limit is reached.

*Marathon Mode:* A continuous Tetris game, that lasts until the player clears a certain number of lines. Generally, the goal is to score the most and stay alive throughout the fifteen levels, and clear all the lines. The game ends when topping out occurs.

*TetriminosTM:* a geometric shape composed of four squares connected at right angles to one another in the pattern of the letter L, J, T, S, Z, or O. (See: <http://en.wikipedia.org/wiki/Tetromino>)

*Key Mappings/Bindings:* The ability to change the default key controls to play the game to a user’s preference.

*Rotate: TetriminosTM* can rotate 90 degrees clockwise or counterclockwise.

*Soft Drop:* The current *TetriminoTM* drops down at a steady pace until until it comes in contact with another piece under it or it reaches the bottom of the game area.

*Hard Drop:* on a certain key press by the user, the *TetriminoTM* drops more rapidly down to the lowest position in the game area it can.

*Tick:* a discrete unit of time that causes a falling *TetriminoTM* to drop down one row in the game area.

*Topping Out: TetriminosTM* stack up to the top of the playing field.

*A “Tetris”:* Clearing four lines at once.

**Requirements:**

1. A single user can play a single player game of Tetris in the following modes:

1.1 Classic [1]

1.2 Timed [1]

1.3 Marathon [1]

1. Two users can play a multiplayer game of ‘vs’ Tetris on a local machine in the following modes:

2.1 Classic [1]

2.2 Timed [1]

2.3 Marathon [1]

1. Score Persistence:

3.1 High scores for each mode are persisted using a SQLite database [2]

1. Generate TetriminosTM

4.1 Generating and using the traditional TetriminosTM [1]

4.2 Generating and using new unique shapes along with the traditional TetriminosTM [2]

4.2.1 Game defaults to only having the traditional TetriminosTM and the user has the option to enable the custom shapes.[2]

1. Game State Persistence:

5.1 A single player user will be able to save their game state, close the program, and come back to it at another time with their progress still in place. [3]

5.2 Two users will be able to save their multiplayer game state, close the program, and come back to it at another time with their multiplayer progress still in place. [3]

1. Key Mapping:

6.1: Allow users, both single and multiplayer, to change the key bindings. [3]

6.1.1: Multiplayer users will not be able to have the same key mapped to the same key another player has mapped. [1, if 6.1 is implemented]

7. Miscellaneous:

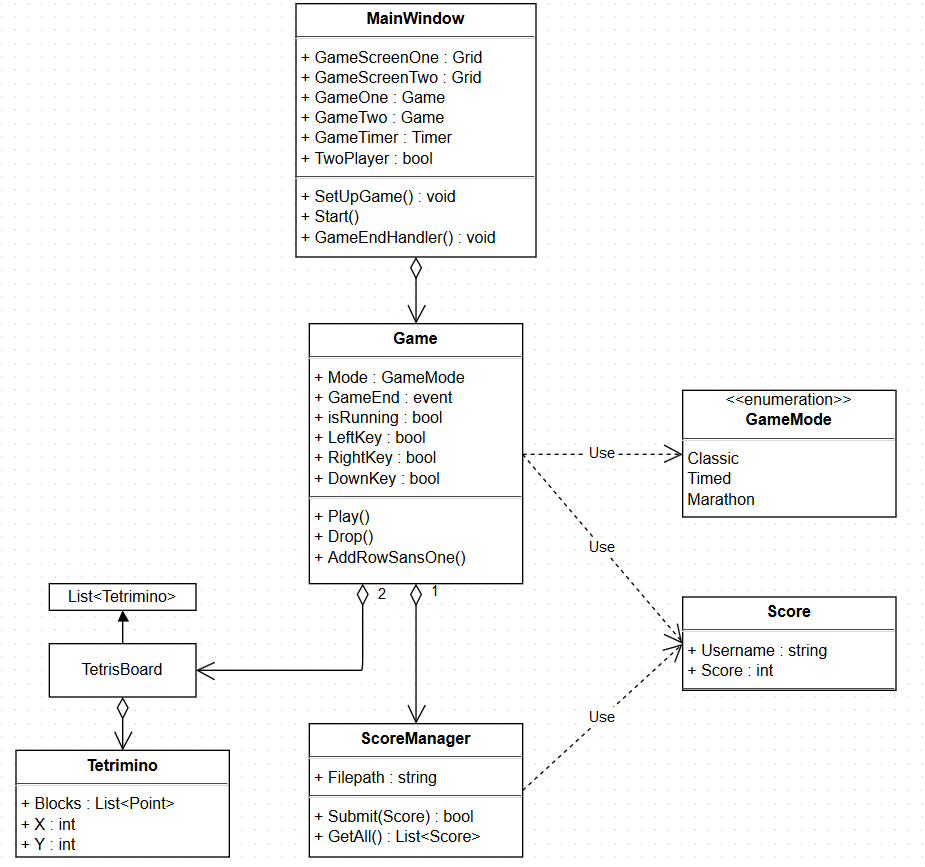
7.1: Tetrismusic plays during game [3]

7.2: Display lines completed [3]

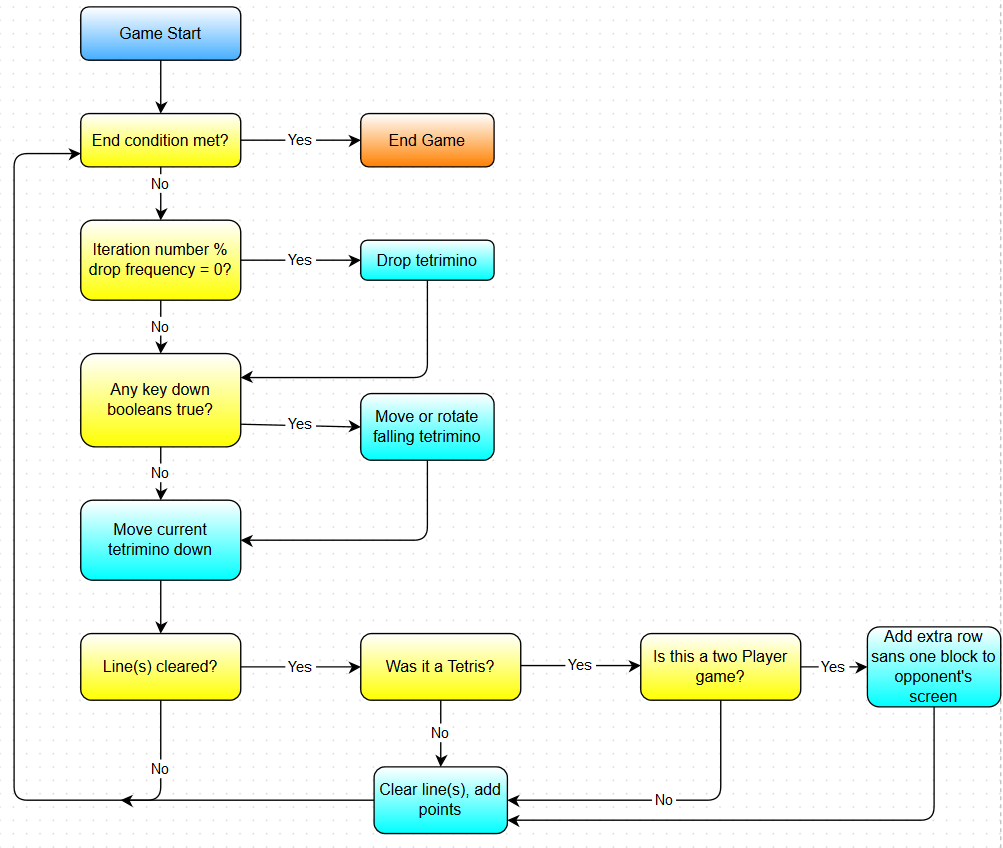
7.3: Speed option for Marathon [3]

**Part 3: Basic Design**

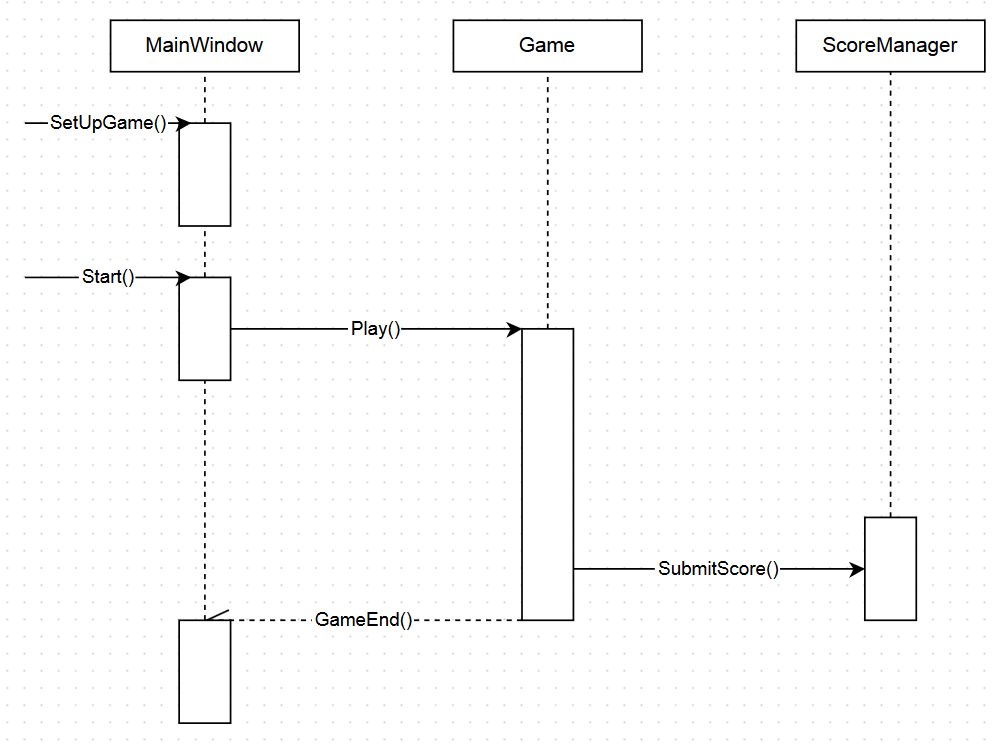
**-UML:**



**-Logic flow:**

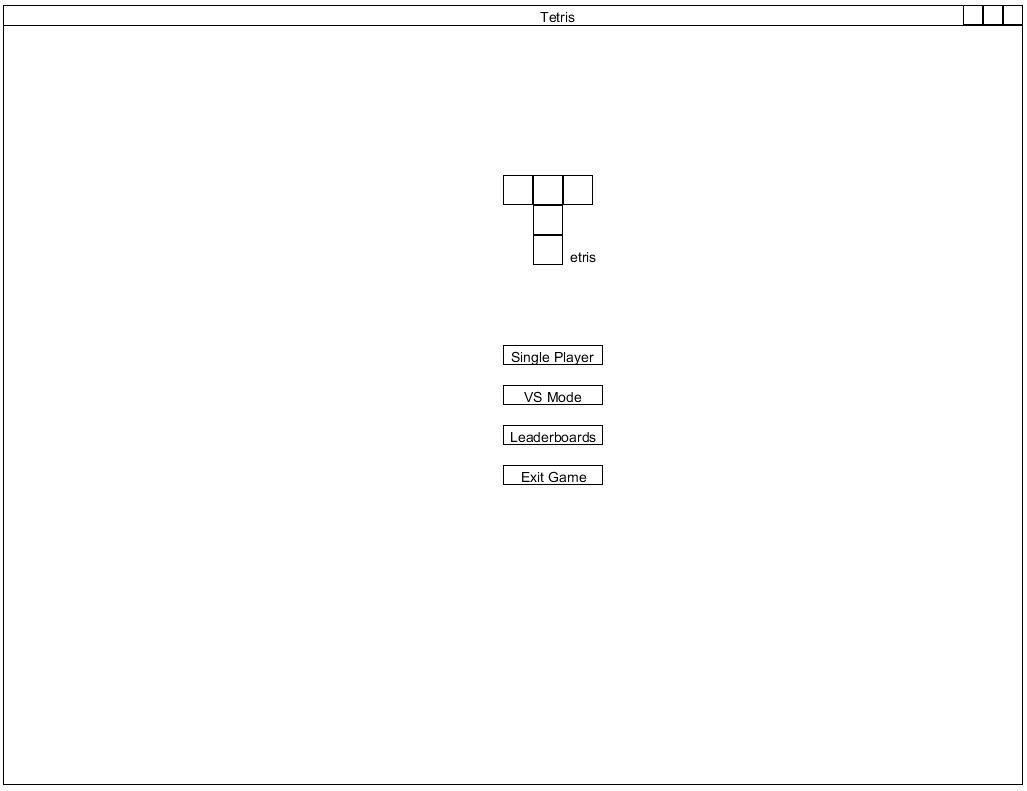


**-Sequence diagram:**

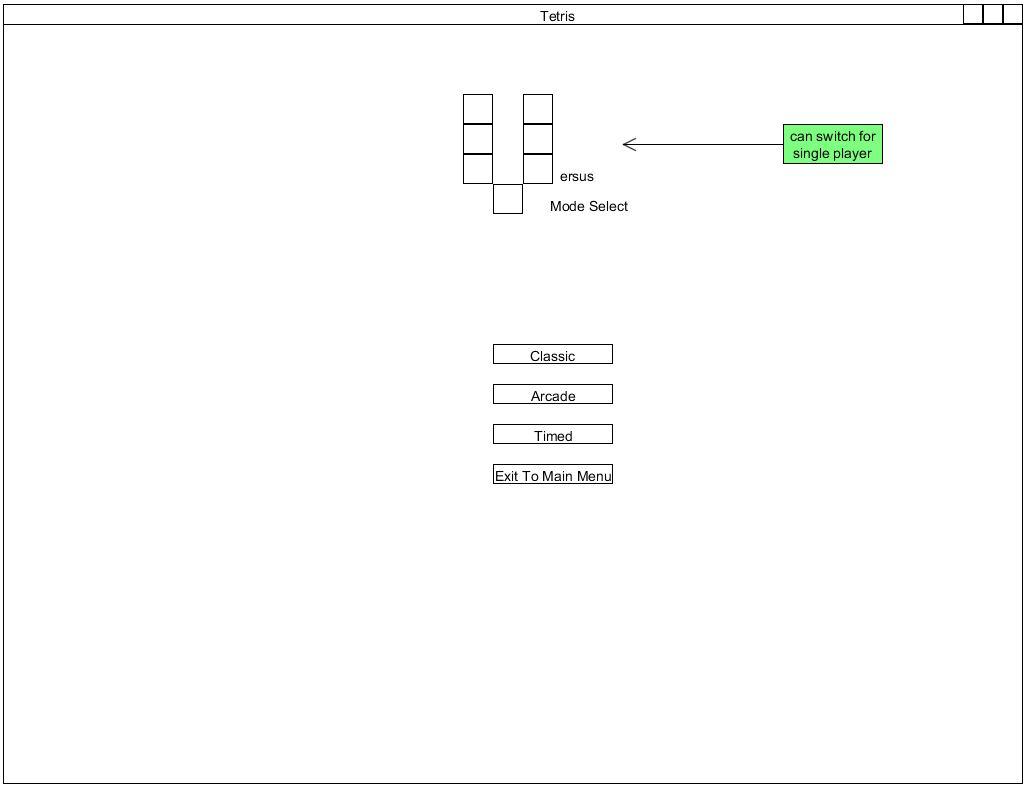


**Wire Frame:**

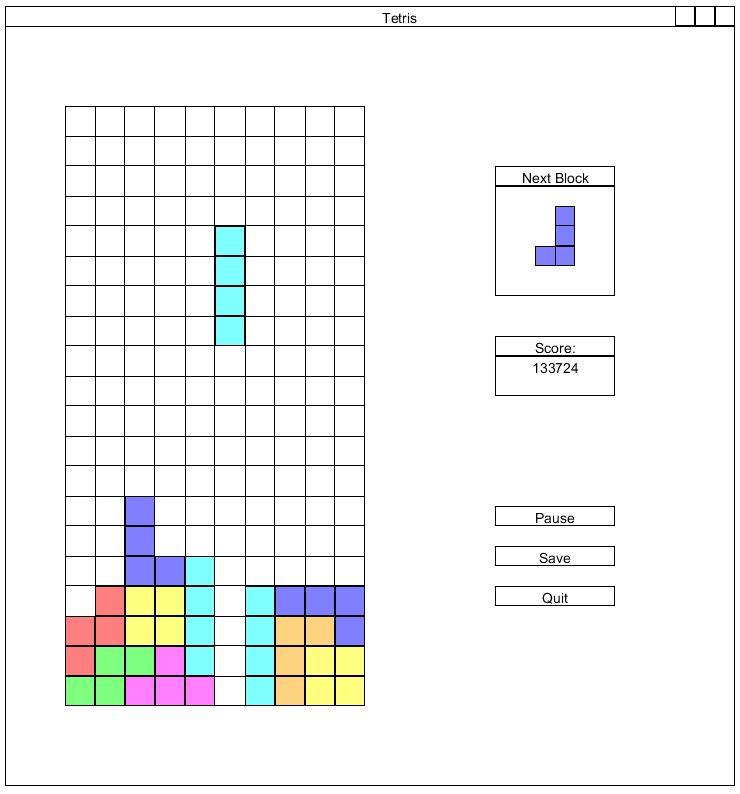
-Main Menu:



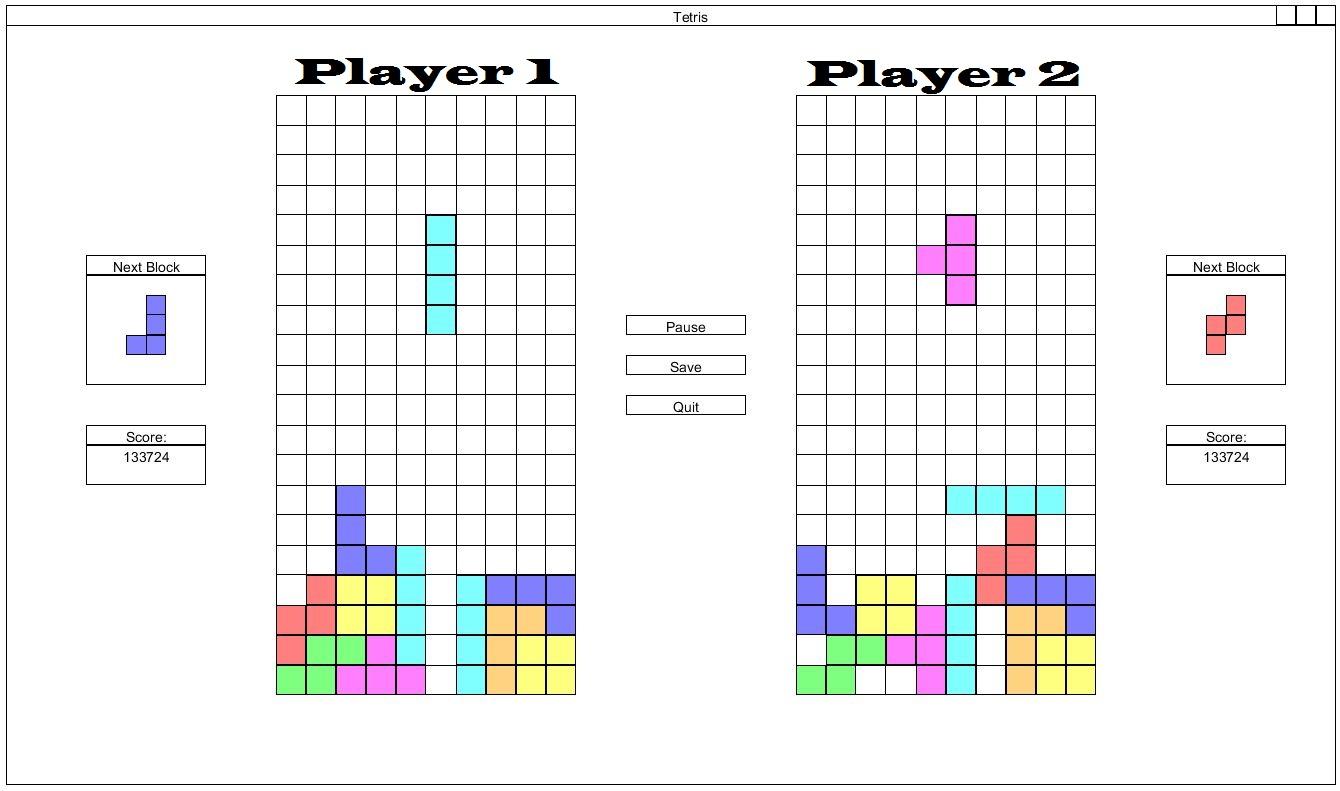
Game Mode Choices: The game mode menu for versus and single player will be the same except for the name/logo at the top.



Single Player: Game play for single player.



Multiplayer: Game play for two player mode.



**Part 4: Project Plan**

Phase 1:

* Create the basic menu UI in WPF (Visual representation only)
  + Scott
* Create the logic behind the three game modes
  + Alex
  + Derek
* Create the standard TetriminosTM
  + Cody

Phase 2:

* Complete ‘functional’ UI for “single-player” menus games (all functionality exists, with or without styling)
* Complete single-player implementation
* Begin high score persistence
* Begin multiplayer logic
* Begin multiplayer UI implementation

Phase 3:

* Complete high score persistence
* Complete multiplayer logic
* Finish multiplayer UI implementation
* Implement custom shapes

Phase 4:

* Finish fully functional and styled UI for all game modes and menus
* Implement option to disable custom shapes, if time allows
* Implement game and menu music, if time allows
* Implement game saving, if time allows
* Implement custom key mapping, if time allows

Phase 5:

* Polish