# Match game Feasibility Study

## **Problem definition**

The **Match** game is a tile-matching puzzle game that I’m planning to build in Java. The goal of the game is to match tiles with the same design to clear the board. It’s aimed at players who enjoy casual, brain-teasing games that are easy to pick up but challenging to master. My plan is to focus on making it user-friendly with simple controls, a clean and colorful design, and smooth gameplay. I also want to include different levels of difficulty to keep players engaged and learning as I develop my programming skills in Java.

## **Problem Analysis**

a. hardware required to run the application

The **Match** game will run on any computer with basic hardware. It requires a modern processor (like Intel Core i3), 2 GB of RAM, minimal storage (around 50 MB), integrated graphics, and a screen resolution of 1280x720 or higher. A keyboard and mouse are needed for controls.

b. User training

The **Match** game will be easy to pick up with a quick tutorial and helpful hints during gameplay. No special training needed—just start and have fun!

c. recommended age of the user

The recommended age for playing the Match game is 8 and up.

d. time the user will need to learn how to use the program

Users will need less than 5 minutes to learn how to play the game.

e. costs associated with developing the software

f. costs associated with developing the software;

The development costs for Match will be minimal, mainly using free tools like NetBeans Java and open-source libraries. The only potential costs could be hosting

g. integration with other software

The Match game will be compatible with both Windows and Mac environments, as it’s built using Java, which is cross-platform.

h. research requirements

I have a solid understanding of the basic concepts needed to develop the Match game, such as Java programming, game logic, and user interface design. However, I may need to research more about handling game states, and exploring libraries animations

j. time constraints

I’m currently busy studying in school while working as a software developer for Property Vista, so I’ll need to manage my time carefully to complete the Match game project. My goal of having a working version ready for testing within 2-3 weeks.

## **Software Project Plan**

### **Subtasks**

* Creating a Project
* Game Interface & Style
* Game Board Grid
* Levels
* Passing Data to Game Components
* Selecting Squares
* Countdown Timer
* Game Animations
* Game Play States
* Switching Game States

### **Pseudocode**

* **Start the game**:
  + Initialize an empty game board.
  + Set the number of moves to 0.
  + Set the number of matched tiles to 0.
  + Set the game status to "starting".
* **Display the game board**:
  + Show the board with hidden tiles for the player to select from.
* **Wait for the player to select a tile**:
  + When the player clicks on a tile, reveal it
  + If it is the first tile, store its position.
  + If it is the second tile, store its position and check if it matches the first tile
* **Check if the two selected tiles match**:
  + If the tiles match, remove them from the board.
  + Increase the matched tiles count.
  + If all tiles are matched, set the game status to "won" and show a victory message
  + If the tiles do not match, flip them back after a short delay and increase the move count.
* **End the game**:
  + If all tiles are matched, show a "You win!" message and the total number of moves.
  + If the game reaches the maximum number of moves or other end conditions, show a "Game Over" message
* **Ask the player if they want to play again**:
  + If the player chooses to play again, restart the game
  + If not, end the game

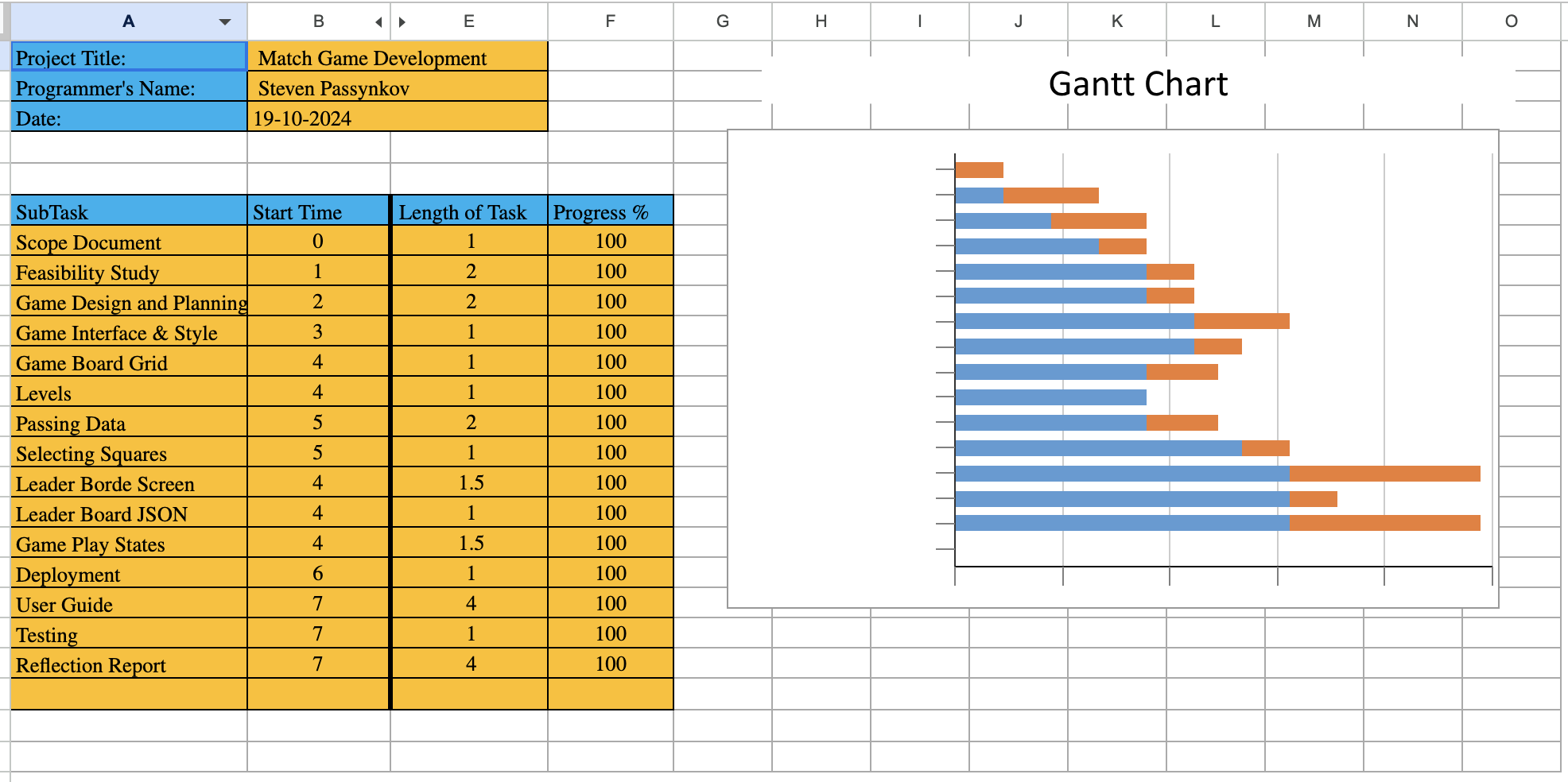
### **Storyboards**

TODO

**Resource list**

* Development Tools - NetBeans Java IDE
* Computer - Computers running MacOS
* Testing - Computers running Windows
* Web Hosting - My try to host on github pages using **jDeploy and CheerpJ.**<https://www.youtube.com/watch?v=mr6VZRrsKAc>
* Mentor - OVS Teacher
* Test Users - friends and parents
* Documentation - Notions and Google Docs

### **Project schedule**



### **Risks**

* Time Constraints - Due to school and work
* Difficulty with Design
* Bugs and Technical Issues
* Low Tester participation
* Github Hosting Problems