Updated

ENGR302 Plan and requirements

Group: S1 - Patrick Mills, Gabrielle Baker, Hanning Qu, Krystal Ng, Max Buller, Sune Weyers

Game name: Papa's Kanbanaria

Delivery date:

- Planning and requirements: 26/07/2024
- Progress report: 16/08/2024
- Design Review: 04/09/2024

Project Objective:

Our group will produce a multiplayer *Papa's Pizzaria* [1] inspired cooking game where players will work as a team to make and sell pizzas in a busy restaurant. This game's purpose is to teach first year computer science students how to use Kanban boards and why we use Kanban boards in a team.

Gameplay:

There will be four different stations that the players can freely move between; service (player takes customer's order), preparation (player prepares the pizza for baking), baking (player will bake pizzas, ensuring they won't burn), and decoration (player will add any final toppings and cut the pizza).

Next Step:

- Based on agreed and MSCW prioritized requirements we create issues on GitLab and start assigning tasks related to further planning.
- Explore different technologies required for the project and complete a UX and architectural design for the game.

Requirements

MSCW Guide:

Must Have [M]: A requirement that must be met for the final product to be considered "complete. It is either necessary for the game to run or necessary to meet the stakeholders' requirements.

Should Have [S]: A high priority requirement that would reinforce the purpose and effect of this project.

Could Have [C]: Would make the gameplay more streamlined and/or would result in a more enjoyable game.

Won't Have [W]: Not a priority to be added into the project, could be additional. The lack of these won't hurt the project's final quality. We would only add these features if we had an abundance of spare time at the end.

Functional

Must Have: Must be done by the end of Week 9.

- The game must be multiplayer and support up to 4 players without performance degradation. The players join the game through different devices over the internet.
- Kanban board will be present from round 1, the user can move tasks around the Kanban board from column to column by dragging the task with their mouse.
- Users can move between stations using clickable buttons.
- Players can perform actions on the pizza (move it between stations, put toppings on it, bake, etc.)
- Needs to have in-depth detail about Kanban boards with emphasis on their ability to:
    - Divide and Allocate work between team members.
    - The importance of review and continuous improvement.
    - The importance of always displaying a task's stage (to do, in progress, etc…)

Should Have: Must be done by the end of Week 11.

- Players can move between the restaurant stations (service, preparation, baking, decoration) throughout the game.
- Customers can take back pizzas if they're not happy with it. Players must correct it based on the customer's request.
- Users can create an account or session using some sort of login and can log in and out securely.

Could Have: Might be implemented if we have extra time.

- Users can select game modes and preferences.

Won't Have:

- Players must be able to track their progress, earn and view achievements/badges.
- Save game statistics and allow the user to view this.

Non-Functional

Must Have:

- AGILE demonstration needs to provide accurate information about chosen Agile programming feature (Kanban boards).
- The game must be compatible with Windows.
- Screen dimensions: the game will have a specific screen size that the group will agree to.
- Needs to have in-depth detail about Kanban boards with emphasis on their ability to help visualize work, limit work-in-progress, and maxim Aptos (Body) ize efficiency.
- The game will have multiple stations available within the team, e.g. putting toppings on the pizza base station, a baking station, cutting the pizza station, and a service station.

- A Kanban board will be displayed on the screen.
- Must be able to be played on local host.
- The players can move freely throughout the stations, allowing them to experiment with different combinations of roles in the restaurant and allocating different numbers of people to each station to find a balance that works.
- User interface must be intuitive and easy to navigate for new users.

Should Have:

- The website code should be hosted on a hosting service, preferably a free one e.g. AWS.
- All user data must be encrypted.
    - Note: we are not sure how much user data we will have at this point, but if there is user data e.g. logins, cookies, etc., there should be some level of security in place.
- The game must handle failures and have failover mechanisms. For example, exception handling in Java should be implemented if an exception is thrown.
- Pizzas get negative points if they are incomplete. If they are complete but not to full quality, they will earn some points but not full points.
- Another player must review the other player's pizza before serving it. This should be in a review station. If another player does not review it, the game will prevent the pizza from being served.
- Game should include scoring, timers and other mechanics relevant to the game.
    - For example, it should keep track of points the players have scored (as a group).

Could Have:

- Tutorial: the game will provide tutorials and help sections.
    - Such as a section explaining how to play the game in the beginning, explanations for what Kanban boards are and their application.
- The user interface should be aesthetically pleasing. A suitable layout, fonts and colors should be selected.
- Implement a database using Firebase, with Google authentication to authenticate users into the program.
- Implement measures to prevent cheating.
- Websites should be relatively safe from malicious attacks. **Won't Have:**
- Game should be responsive and must resize for all screen sizes. (May not be necessary.)
- Large variety of pizza flavors and toppings.
    - This may not be achievable in the timeframe and would not help reinforce the purpose and effect of the game.
- Large multiplayer server with large numbers of players (e.g. the entire class in the same game.)
    - This may not be achievable in the timeframe and would not help reinforce the purpose and effect of the game.
    - May help with this being used as a teaching tool, however, with large numbers of people the game mechanics might become too complicated
- Implement 3D graphics.
    - Just implementing a 2D game will be complicated enough. 3D would likely be too ambitious for the time frame.
- The game could be compatible with major operating systems and web browsers (online): Chrome, Firefox, Edge, Linux and Mac.

Project Risks and Mitigations:

Risk 1: We fail to complete the base game by the deadline and submit an incomplete game (*Must Have* requirements not met).

Mitigation 1: This can be mitigated by having a plan and regularly carrying out group meetings (usually Tuesday and Friday) to make sure we stay on track and on the same page.

Mitigation 2: The Must Haves must be a minimum viable product (MVP) that can be done within ~50% of the remaining time allotted.

Risk 2: The team becomes dysfunctional, leading to the project not being finished on time or finished to a low quality.

Mitigation 1: This can be mitigated by holding regular group meetings, talking about the code and planning our work together.

Mitigation 2: Identify the root causes, engage in open and honest communication, clarify roles and responsibilities and identify any additional resources or support the team might need to succeed by offering guidance

Risk 3: A team member becomes unavailable to work for whatever reason.

Mitigation 1: Our Must Have tasks have been carefully selected so they can be done by a smaller team within the allotted timeframe.

References:

[1] https://www.coolmathgames.com/0-papas-pizzeria

[2] https://www.agilebusiness.org/dsdm-project-framework/moscow-prioririsation.html

[3] https://www.agile42.com/en/agile-teams/kanban-pizza-game

[4] https://www.atlassian.com/blog/productivity/how-to-write-smart-goals