Game Development Postmortem: BizWorks

Studio Calibrate

Kimberly Huang

Adela Kapuścińska

MinSun Park

Toya Rosuello

May 10th, 2018

BizWorks

Table of Contents

Description

BizWorks is an educational financial literacy game that uses elements of business simulations and resource management to explain financial literacy concepts. It is a crossover between a business simulation game, role-playing game and party game. It has been released on the iOS and Android.

In the game, the player is a bakery owner who has taken out a loan to start their business. The player manages their employees' week to week schedule and budgets their earnings for various expenses including stock, employee salaries, loan payments, insurance payments, and tax payments. Ultimately, their goal is to pay off the loan that they initially took out, all while giving financial advice to their employees based on each employee's particular story.

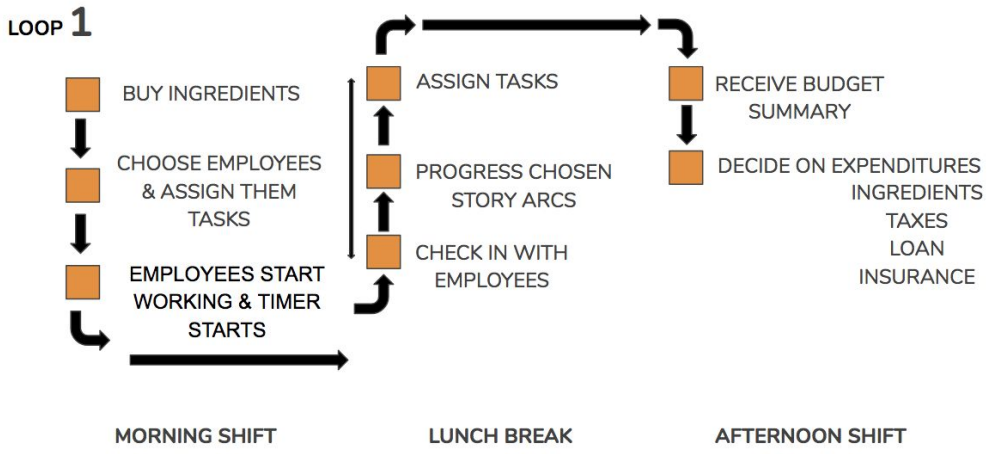
Educational games tend to fall flat with their target demographic as a result of how gamified they are; they are usually predictable in their aim to educate, and do not innovate on the video game medium besides wanting to capitalize on it. Therefore, players usually end up playing the game, but derive no real enjoyment or learning from it. We were interested in engaging players by virtue of a management challenge, as well as by creating compelling narratives that the players could feel invested in or relate to. We hoped this would help players remember the financial literacy topics taught and give them a concrete idea of how the lessons can be applied to their own lives.

Development Stats

- Developer: Studio Calibrate
- Publisher: Studio Calibrate
- Release Date: May 4th, 2018
- Length of Development: Semester (4 months)
- Number of Developers: 2
- Number of Artists: 1
- Development Tools: Unity
- Creative Tools: Photoshop, After Effects

GamePlay

Overall Mechanics

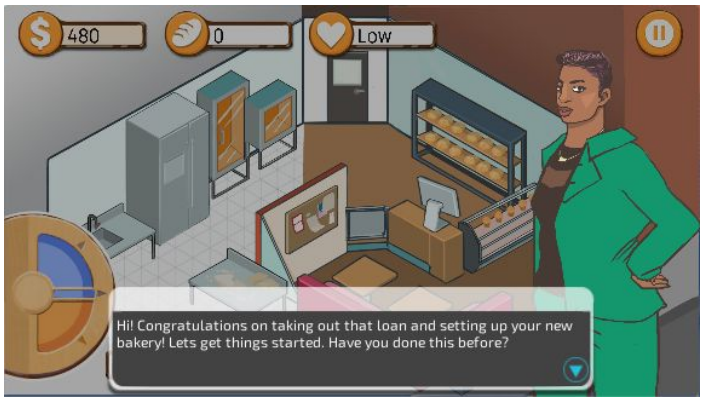


Individual Features

1. Game Start Screen

2. Tutorial

If pursued, the financial advisor walks you through the first game loop of the game, which explains the primary and secondary mechanics of the game.



3. Buy Ingredients

Every **month**, players can stock up on the ingredients that are used to make the baked goods. The **PREDICT** functionality was implemented so the player can get a sense of what to expect for the following month.

4. Choose employees

Every **week**, players choose two employees to work. Different employees have different specialties and morales, the latter of which is influenced by their financial decisions.

The player then assigns two tasks per employee by dragging them into the blank spaces. When the employees are assigned, they start working immediately.

5. Morning & Afternoon Shifts

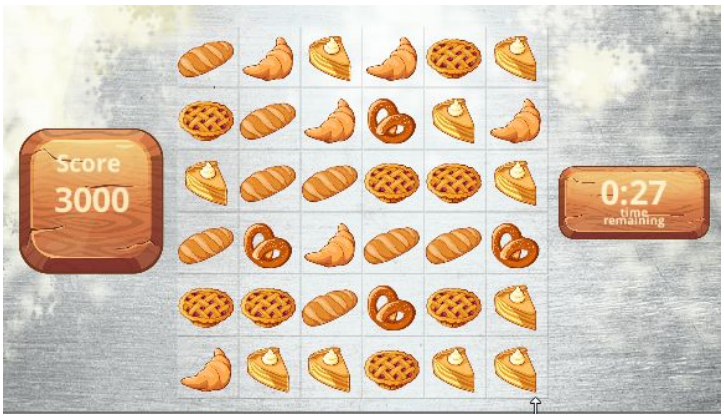
A week is comprised of a morning shift, a lunch break, and an afternoon shift. This information is indicated with a UI element on the left in each phase. Each shift lasts 30 seconds, which is also indicated with by a UI bar that counts down on top of each employee.

Depending on the tasks assigned, the baked goods are either SOLD or MADE. This is shown to the player in a few ways:

- The employees assigned to SELL are in front of the cashier, while those assigned to MAKE are at the worktable.
- Every time an employee sells or makes a product, a small UI blinks on top of the employee that says “$+1” or “Bread+1,” and the amount gets added to the total product count or money count on the top UI bars.
- When the baked goods are being sold, the customers show up at the store.
- The current bread count is shown on the top UI, as well as on the bread stand in store.

6. Mini-game

As a way to increase replayability, especially considering that this is an educational game that could seem boring to the target demographic, the team incorporated a mini-game into the main game that can be accessed through workstations.



The players play the mini-game in order to accelerate shift time and earn extra money for their business. The concept here is that the player is helping the employees, which is represented by the mini-game. The mini-game starts when the player clicks on a worktable or a cashier.

The mini-game in this prototype is a MATCH-3 game that lasts for 30 seconds. The score the player earns is translated into an amount of bonus money, which is transferred into the main game.

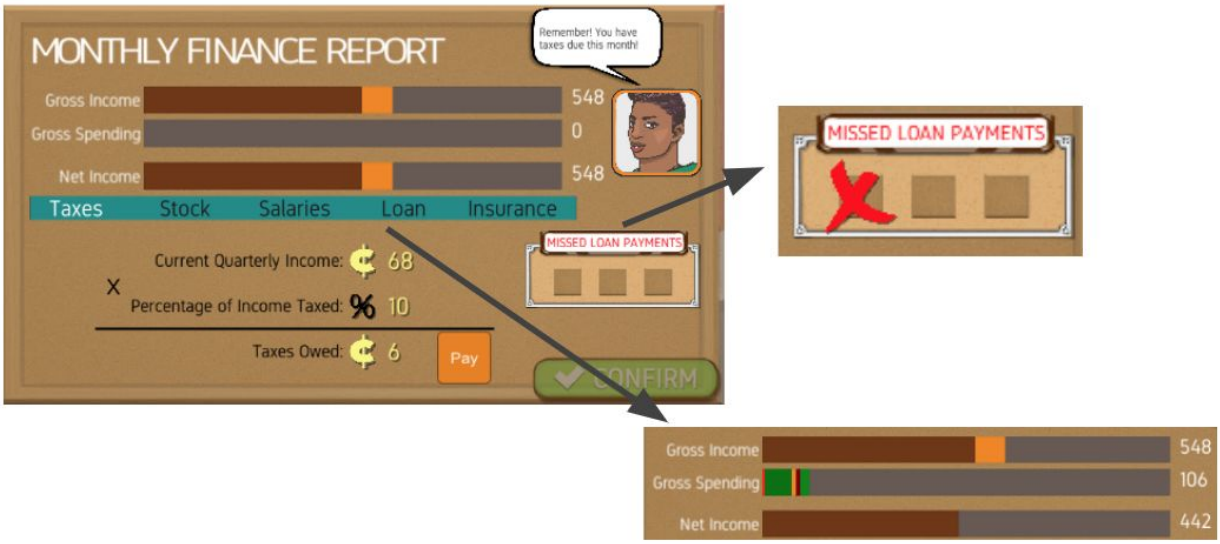
7. Lunch break

As previously mentioned, a main part of our game is the interaction between the player and the employees. While the players can interact with the employees during the shifts, a lunch break is where the main interactions happen.

The purpose of the interaction is to let the players (students) practice taking responsibility in their decisions without giving them too much pressure, as requested by the client.

The conversations for total 4 characters were written, and were implemented in the game using Fungus plugin.

8. Monthly Summary



After four weeks in the game, which adds up to a month, a Monthly Finance Report panel shows up to give the players an idea of how well they performed. The report shows how much the player has earned (Gross Income) based on how much they've sold and delivered, as well as Gross Spending, which is calculated based on the tax payment, money spent on ingredients, salary for employees, and other costs. There is the option to pay for the loan with penalties if the player underpays or fail to pay when payment is due and insurance payments for the business.

What Went Right

Successful Prototype (Good Proof of Concept)

Having defined the cross-over of elements of *BizWorks*, the team had a very concrete idea of where the game would be headed and what we wanted it to be. This allowed us to stay on course during the development of the game, as well as be efficient in re-scoping the project when needed, understanding what had to be cut and what had to be kept to still achieve an experience that reflected our initial goals. The features were cut in a way that allowed the team to see how they could be received when implemented in an expanded prototype. We were able to achieve a good balance between the elements of a business simulation game, role-playing game and mini-game; *BizWorks* has a good flow and feels cohesive. Ultimately, the team was able to build a prototype that may have been limited in scope and features, but showed that the rationale behind different mechanics in the game was on the right path.

Cohesive Art Style

Given the premise of our project, our aim was to make a sleek, modern-looking mobile game that feels good to play and that plays well. We aimed for a style that would be recognizably comfortable, bold, minimalist (and so allowing for future customization), and illustrative in order to garner interest, especially narrative interest. Elegant, simple animations complete this stylization and add a nice touch to the overall feel of the game. Early-stage playtesting proved that the theming wasn't as important as narrative, so we pursued a bohemian city fantasy, which was something more realistic than fantastic.

We wanted to emphasize the idea that the player is very much a part of the microcosm that is their bakery. For this reason, we decided on an isometric view– to focus on the inner workings of the business, as well as help the player visualize the space and themselves inhabiting it. What's more, we found that this perspective generated more visual interest than if it had followed the head-on perspective of existing simulation games. This also gave us the opportunity to make a clear distinction between the main game hub and the lunch break area, both architecturally and mechanically. A door connects the two and, with warm light streaming from the outside, promises some leisure time in the game.

We understood that sleek, comprehensible UI was going to be a necessity for navigating the various features of *BizWorks*. Orange was established as the primary color of the game, serving to indicate major UI elements and to give overall visual cohesion. To further strengthen the relationship between the player's tools and their tools as the bakery owner, we pushed for environmental interactions and diegetic UI. As an example, the player can access the centrally placed Managerial Board to return to the overview of stock, employee selection and task assignment that they had interacted with at the beginning of the week. It was also important to establish consistency between the diegetic UI and in-game world: the delivery bike appears both as an active UI element (if an employee had been tasked with delivery) and stands idly in the lunch area (when not used).

Essentially, almost all art choices were made very deliberately, down to the color of the text and highlights on the appliances. These choices were made in an attempt to keep the appearance of the game appealing and keep players invested in playing the game. Previous prototype iterations in the process

showed that without the visual elements tying the whole game together, the game felt boring and choppy and did not grab the attention of the player. Therefore, being able to end with a visually cohesive final prototype was a large success in making the game feel better to the player.

Mini-Game Integration

An important element to *BizWorks* is the mini-games. These serve as a way to engage the player, as well as provide a narrative shell for the 'work' that the player performs as the bakery owner during their employee shifts. Mechanically, the mini-games also earn additional money and speed up the waiting time for the tasks to be completed. For the purpose of the prototype, we were able to implement one mini-game, Match-3, that could be accessed through the worktable and cash register. The hope was to have a selection of mini-games that would be themed based on the various affordances of the bakery (sales, delivery, stocking, making dough, baking, washing dishes) to diversify the player experience. In the final prototype, the team was able to make a mini-game that flowed cohesively from the main game.

Diverse Narrative Portfolio

There is a lot of narrative potential in the story element of *BizWorks*. The team designed the characters to be within the similar age group to the target demographic, as a way of making them more relatable and their financial considerations relevant to the players. However, each employee also had their own unique set of financial issues, making them diverse in the lessons they covered. During financial turning-point conversations with the employees, we offered 3 decisions: the affirmative ("Yes, I know."), the negative ("I don't.") and lack of relevant knowledge ("Let me ask the Financial Advisor."). We found this range of answers important, in that it offers that the player take up the financial inquiry themselves. It also established the Financial Advisor as not only a point of contact in these decisions, but also as an archive of further knowledge. The decisions that the player made for the employee also made an impact on employee production, which was used partly as an incentive for the player to make the right decisions. The team was successful in making the game cover a diverse range of financial issues without imposing a certain viewpoint on the player.

Financial Advisor

The Financial Advisor character, Danielle, is a crucial part of helping the players navigate the game. She starts off as the tutorial character, introducing the player to the mechanics and the narrative behind the game. She also provides a lifeline to players who are unsure of the advice they are giving to the employees, offering in-depth knowledge of the concepts that the employee's narrative explores. The team's goal in including her was to provide a safety net to players who are less knowledgeable about finances. She also existed as another form of feedback for the player. At the same time, because speaking to her is an optional choice, both for the tutorial and during dialogue with employees, more experienced players are not forced to relearn concepts they already know. Employee interaction becomes more of a review in these cases. Finally, establishing her as a significant presence narratively, promotes the idea of bolstering relationships with local businesses and customers. The financial advisor gives the player someone to reach out to if they are unsure about their decisions and basically slides an expert into the game without needing the development team or financial literacy expert present when the game is played.

What Went Wrong

Over-scoping and Slow Prototyping

From the beginning, the team understood the potential that *BizWorks* held and did not hesitate in creating an elaborate, big picture image of the game. However, the game proved to an immense project given the amount of time the team had. Therefore, there were many features that unfortunately didn't make it into the prototype. Some of these features include the store reputation (a variable that would change based on how well you're running the business), more environmental interactions (diegetic UI and store customization), and unexpected events (random events that would hurt the store, which would make the game challenging to play and give mechanical significance to having insurance).

Over-scoping on the project also came with the side-effect of prototyping far slower than intended. The sheer number of mechanics made it difficult to know what to focus on during implementation and the mechanics themselves were not fleshed out well until several weeks into development. This issue resulted in underestimating the time it would take to implement several features, like dialogue and the summary panel. With many mechanical pieces moving at the same time, the team was not able to playtest the whole game early on in the process and could only get feedback on individual mechanics. Focusing on the core mechanics of the game first would have allowed for faster prototyping and smoother playtesting of the concept. The slow prototyping also hurt in getting out a polished prototype, as the final build still has some bugs that the team wished it has more time to look at.

Team Member Changes

In the course of the development process, the team underwent major changes in the team makeup. The assigned project manager withdrew from the course early in the project due to a scheduling conflict, and the supporting artist left a little past halfway through the process due to personal reasons. Both made large contributions to the game design of the project, and their loss not only put back the production of the game, but also greatly impacted team morale. While the team was able to redistribute the work amongst the remaining members, the many moving parts of the game proved to be an exhaustive undertaking to manage, with some members having their workload essentially doubled. These changes necessitated earlier and more efficient organization, which the team also had some challenges with and the lack of organization ties back in with the over-scoping and slow prototyping issues mentioned earlier.

Scheduling/Organization Difficulties

Scheduling for work sessions was difficult due to each team member having vastly different schedules. There were very few times over the week that every member could meet and nearly no times during the weekend, which made putting progress reports together before Monday check-ins with the client more difficult and also held back the development of the game, as not being able to meet meant that there were more potential for miscommunication in how the team wanted certain features to look or feel. While the team was able to circumvent some of these scheduling issues by keeping in constant communication over the internet, it was not as efficient as it could have been if the team could meet consistently for work sessions.

Although the team kept in constant communication, communication occurred over several internet platforms, which necessitated repeating information over every platform to ensure every team member received it. As a result, assets and crucial information sometimes ended up spread over several methods of communication, making retrieving resources at a later date difficult. While the team tried to stick to Slack through the second half of the semester and used Trello to communicate tasks, not all details about any team member's tasks were always expressed, thus leading to some confusion about where the team was in the process. Looking forward, it would be imperative to concentrate all communications on one platform and if teamwork sessions are not possible, have all communication of tasks be as detailed as possible.

Trouble Generating Narrative Interest

While the writers were able to create complex, interesting narratives for each employee, the presentation of these stories was limited by how the dialogue functionality was implemented. The programmers' lack of familiarity with the Fungus API coupled with the API itself being more geared towards dialogue that would be set prior to runtime resulted in a very rigid structure that dialogue had to follow. This structure did not allow for setup prior to employees asking for advice which made these questions seem abrupt and awkward. The disconnect between writing and programming could have been solved with better communication about the structure of the narrative arcs.

In addition to the technical issues, the writing team had to solve the more ethical issue of not pushing an ideology onto players. The team's goal was to create a game that teaches financial responsibility, but we did not want players to feel that they should never follow their goals if their goals were not financially sound to pursue at the time. Some early scripts blatantly told employees to give up on passion projects, which we felt was not the lesson players should take away from the game. Furthermore, with the structure of the dialogue having a "right" answer and a "wrong" answer, writers did not want to create scenarios that would make the player choose different priorities based on a certain set of moral beliefs. For example, although extreme, the team did not want to give the player the choice of either saving money for their parent's surgery or paying off their student loan debt, as the team did not feel like it was the their place to tell the player which was more important. While it may not be perceived in that way, with the writing structured with "right" and "wrong" answers, the team could not present scenarios that made the players choose priorities and wanted to focus on sound financial decision-making. Being constantly aware of this balance, the team struggled in early scripts to make sure they were not pushing a certain mindset on the player and this struggle may have led to less diversity in issues than the team would have liked.

Technology Issues

Mid-development, the programmers ran into several issues with version control that was holding back progress significantly on the prototype. More specifically, Git was unable to handle Unity scenes properly, resulting in two major issues.

The first issue involved setting gameobject references in the scene. Gameobjects such as the GameManager required multiple references to other game objects in the scene and these references often did not persist between pushing and pulling to the repository. This required programmers to set these

gameobjects in the scene each time they pulled from the Git repository which took up a large portion of development time.

The second issue involved Git's inability to efficiently resolve merge conflicts between two versions of the same Unity scene. Because a good portion of the game happens in the same scene, this caused merge conflicts nearly every time a programmer made a pull request. Programmers moved to Unity's integrated version control system, Unity Collab, in order to facilitate development. However, this move came with its own limiting factors, including the programmers' lack of familiarity of the system and an inability to branch versions. However, Unity Collab's ability to properly understand and merge scenes was crucial to faster development.

Playtesting

With the final prototype completed, the team ran some playtests to see how well their objectives were implemented in the game. Since it was difficult to find players of the target demographic ready to playtest, the team targeted freshmen in college, but eventually did allow older playtesters to play to get more results. The bulk of playtesting was done over the course of two days. Players were asked to play the game until they felt like they had a good understanding of the mechanics, with a minimum playtime of one month. After playing the prototype, players were then asked to fill out a post-game survey, which asked for demographic information, as well as detailed information about different aspects of the game.

The overall impressions of the game were mostly positive. 87.5% of users said that the game was enjoyable. However, playtesting revealed clear shortcomings in some aspects of the game. For example, the team decided to use a tutorial to help players go through the first week of the game. However, even with the tutorial, 25% of the players expressed some trouble with understanding how to buy ingredients and assign employee tasks. Part of the issue was that some players ran into bugs while trying to confirm purchases, but others were confused about how the UI was structured and how mechanics like the predict function worked. A problem that was common across all users is that they tried to assign both tasks for an employee during the same shift. Another problem the team found was that most users were unclear about the minigame mechanics until they played the game three to four times. 37.5% of playtesters said that they wished there were clearer instructions for the game and wanted a better explanation about how the minigame tied into the main game.

For the most part, the playtesters enjoyed employee interactions. They did not need to be prompted to click on the employees during lunch and took the initiative to do so. Multiple people mentioned wanting to finish the character arcs of the employees before they finished playtesting. However, only 50% of playtesters felt invested in character storylines. For those that did not, their main problem was that there was not enough employee interactions, so they got bored after plateauing in the storyline. One playtester did mention that the interactions felt fake or forced. As the team was not able to completely implement employee interaction like the fluff dialogue that was created for the prototype, the problem of investment can be solved by implementing all the scripts that were written.

The biggest problem that playtesting revealed was that there was still not enough feedback for the players. Most of the players did not notice the number of stock increasing or the money being earned, which the team decided to show through occasional visual feedback. Beyond that, most players kept assigning employees to bake bread, but did not notice when they were out of ingredients and did not buy more. This issue snowballed into players not having enough money at the end of the month to pay off their loan and resulted in a net loss. Even the bars at the top of the screen did not help some playtesters when the amount of bread in stock stayed at zero. Since the main objective of the game was to teach students budgeting and planning for future events, it is important to make sure that they are made aware when there isn't enough inventory, so the feedback UI needs to be updated in future iterations.

Future Iterations

After playtesting, there are a few obvious changes that the team would make for future iterations of the game. First, a more fleshed out tutorial would be helpful for the players, especially when playing the minigame and looking at the monthly panel. A lot of player confusion came from being thrown into the game right away, so adding to the tutorial could mitigate that confusion. Second, as mentioned, employee interactions would need to be expanded on by including fluff interaction and more story points. Third, the feedback UI in the game could be made more obvious so that the player knows when they are running low on ingredients or when not enough bread is being produced. One way this could be implemented is by having a notification system that pops up a dialogue box saying "You're low on ingredients" near the shift indicator when the player needs to buy more. Additionally, the team took out the sound feedback when the employees produced product. Bringing it back may help players notice when they are not producing stock for the game.

While *BizWorks* is a bakery business game, there is a lot of potential to expand the game beyond what the team currently has, since the team has created the fundamental framework for the game. 62.5% of playtesters thought that the game would be more interesting if the number of businesses increased in the game. Additionally, 75% of players thought that increasing the number of employees would make the game more interesting, while 87.5% of players thought that increasing the number of mini-games would make the game more interesting. While the team could not increase the number of businesses, employees, or minigames in this prototype due to time constraints, future iterations of the game could see a larger variety of those aforementioned features, thus piquing more interest in the player and getting more players interested in the employees and keeping them interested in the game.

Conclusion

Making *BizWorks* has taught our studio the value of rapid prototyping and focusing on core mechanics. This was the most limiting factor during development, preventing us from receiving more player feedback and iterating on our game. Technologically, our studio better understands Unity's workflow and best practices, including making more modular code and loading resources on start rather than during updates or on events. We now have a better understanding of what functionalities Unity is able to handle without extra scripting and what functionalities need more customization for the studio's purposes. Programmers will strive to implement these practices in future projects.

From a development perspective, this project has given the team good insight on some features that can make educational games more enjoyable. The model that we had of combining business simulation games, role playing games, and party games created a game that was interesting to the player and kept them invested, while still teaching financial literacy concepts. It is clear that giving players more creative freedom in how they interact with the game environment, including features like customization and diegetic UI would only increase the amount of interest they have in the game. Teaching financial literacy concepts through character interactions proved to be a good method of educating players, as it could be worked into the game in a creative context. Considering the employee interaction and the game overall were received well by playtesters, Studio Calibrate has come to the conclusion that this type of game is on the right path when creating a game for the sake of teaching financial literacy to high school students.