Stardew Assistant Proposal: A Quality-of-Life Mod

Vision Statement

This project aims to build a Stardew Modding API (SMAPI) mod for Stardew Valley that improves the in game experience by adding subtle but useful quality-of-life features. The mod will include a nightly diary prompt, a smarter shopping interface that shows crop profitability and growth constraints, streamlined seed purchasing, and visual cues for Community Center bundle items. The goals are to help players make more informed decisions, remember their progress, and reduce tedious tasks, without changing or breaking the balance of the game.

Motivation

As a relatively new Stardew Valley player with other obligations and a tendency to play with an eye toward optimization, I often pause my game for weeks and return unsure of what I was working on when I last played, leading to wasted time. I've also found the in-game shop lacking info when it comes to making strategic, informed farming decisions. This mod addresses these frustrations by enabling players to leave themselves notes, avoid bad seed purchases, and better track bundle requirements. Apart from personal motivation, I hope the project will demonstrate practical experience in game modding and/or working with an established code base, UI integration, data parsing, and user focused design. While the project is specific, these skills should apply more broadly in software development.

Planned Features

This mod will add three new features to the game, each designed to improve the player's experience without changing the balance or difficulty of the game. All features are modular and can be developed independently of the others, which supports flexibility during development.

1. **End of Day Diary Prompt**
   At the end of each in-game day, the player will get a prompt asking if they'd like to leave a note about what they worked on that day or what they are working towards in the long term. Entries will be stored per save file and can be viewed later from an in-game menu similar to the quest journal menu. This is completely optional, but is intended to capture more custom detail than the quest journal and help players remember their goals between play sessions.
2. **Shopping Assistant at Pierre's**
   When the player opens the shop menu at Pierre's, quality of life improvements will be added to the menu to help with crop planning and purchasing:
   - Crop Warnings: If a crop's growth time exceeds the number of days remaining in the current season, the crop will be visually flagged with red text or a tooltip.
   - Profitability Display: Each seed will display estimated gold-per-day or gold-per-season based on base sale prices. Typically, the player only sees the

    seed price and growth time to maturity, so this will give the player pertinent info to make more informed decisions.
  - Easier Bulk Purchasing: A simple UI will allow the player to buy a specified number of seeds without clicking repeatedly or clicking and holding the mouse.
3. **Community Center Inventory Markers**
   While viewing their inventory or a chest, the player will see a small marker on any item that is still needed for an unfinished Community Center bundle. This only applies to items that haven't yet been donated to the Community Center. The intention is to reduce the need for players to repeatedly check the Community Center screen while gathering items to donate.

Learning Objectives and Time Scope

This project is designed and intended to build practical, portfolio worthy experience in game mod development, interface design, C# programming, and working with large, complex codebases like Stardew Valley's. Over the course of the project, I'm certain there will be unexpected rabbit holes and challenges, but at the very least, I will gain experience in the following:

Learning Objectives

- Learn how to build, compile, and load a functioning SMAPI mod in C#. With this comes experience in event driven programming in a real codebase.
- Understand and manipulate game data structures to create responsive features that reflect the game state in real time. This will require parsing data like crop metadata, player inventory, and Community Center bundle progress.
- Design and test intuitive (hopefully!) UI components using SMAPI's UI APIs. This will allow features like the end of day diary prompt, profit indicators while shopping, and visual items markers in the player inventory.
- Modify existing game menus, such as in Pierre's shop, to add dynamic tips, visual warnings, and purchase improvements.
- Practice data persistence by saving and loading mod specific information (like diary entries).
- Publish the project on GitHub with documentation, code comments, and a README that allows others to understand, use, and extend the mod.

Time Scope

Each feature of the mod is independently testable and can be implemented in stages, which makes it easier to log progress and stay on track and within scope. While I expect some surprises and unforeseen issues, I hope the breakdown below is as accurate as it can be at this stage.

- SMAPI setup, gaining familiarity, and proof of concept with basic mods: ~6 hours
  - Initial setup and confirming access to the game environment, getting a minimal mod to compile and run.
- End of day diary prompt system: ~10 hours

    - Create a prompt for the user at the end of the game day, saving text entries per game day, and building a UI to view past entries
- Shopping Assistant for crop warnings and profit info: ~12 hours
    - Reading crop data, comparing growth time with days left in the season, calculating or retrieving profit margins per crop, and integrating tips or visual cues into the shop menu.
- Easy bulk seed purchase: ~6 hours
    - Extending the purchase system to include a text field that allows buying a specified number of seeds at once without repeatedly clicking or clicking and holding.
- Inventory and bundle tracking indicators: ~10 hours
    - Parsing the player's inventory and active bundles to enable marking items needed for a bundle in the inventory or chest with a visual.
- Final polish, testing, and documentation: undetermined
    - Fixing bugs, refining UI, writing README with usage instructions and more.

This plan leaves room for adjustments as the project develops and changes. Since the features are modular, it should be easy to adapt or shift direction if necessary.

Risks to Project Completion

While generally full of unfounded confidence, I'll admit there are some risks that could impact project completion:

- Learning curve with C# and SMAPI:
    - I've never worked in SMAPI's modding environment in this capacity before, but I have some exposure, I have installed mods, but I haven't created them before. I'm also unfamiliar with C#, but I've used C++ and even C before, so hopefully this won't impede progress too much.
- UI design complexity:
    - Making clean and functional in game UI additions might take more time than expected, especially given my lack of experience and limited documentation and examples.
- Bundle tracking logic:
    - Accurately identifying which items are still needed for Community Center bundles may be more difficult than anticipated, especially if state tracking is inconsistent and/or incomplete.
- Time management and scope creep
    - With multiple independent features proposed, there's a risk of spending too much time polishing one and not leaving enough time for the others.

Mitigation Strategies

To address the above risks, I plan to take the following steps:

- Start small and ramp up:

    - I’ll start by creating a minimal “hello world” equivalent SMAPI mod to gain familiarity with the structure, event system, and C# before starting larger feature development.
- Start with simple UI elements and reuse existing examples
    - I’ll avoid overly custom UI unless time allows, and start with simpler UI such as tooltips, basic prompts, and overlays.
- Use official game data and modding APIs
    - SMAPI provides access to some save state variables, like BundleData, which will help to avoid manually recreating bundle logic from scratch.
- Follow a modular, feature based timeline
    - Because the planned features are independent, I’ll schedule development week-by-week and focus on completing one feature at a time. If time runs short, I can skip or simplify lower priority components without derailing the whole project.

Project Assessments

I will consider the project to be successful if it meets the following criteria:

- Functionality: All core features (end of day diary prompt, shopping assistant, and inventory bundle markers) are fully implemented and behave as intended.
- Reliability: The mod does not cause crashes, bugs, or save corruption during typical gameplay conditions. Diary data entered by the user is saved and loaded correctly across sessions.
- Usability: Features are easy to discover and use. In game prompts, tooltips, and UI elements are clear and intuitive.
- Code Quality: The project code is well structured, clearly commented, and version controlled in a public GitHub repo.
- Presentation: The public GitHub page includes installation instructions, screenshots or demos, and a brief overview of each feature, making the project accessible to all users.

Project Portfolio Link

The project will be documented and available at:
GitHub: https://github.com/BenLittle24/Stardew-Assistant