Expense Tracking System

Ashish Kumar Singh, Manan Batra
Chandigarh University

Contents

1 Product Introduction

Introducing Budget Patrol, your personal finance companion right in your Chrome browser. With Budget Patrol, managing your expenses has never been easier. This intuitive Chrome extension empowers users to take control of their finances by creating customized spending categories with allocated budgets.

Simply set up your spending categories based on your financial priorities, whether it's groceries, entertainment, or savings goals. Allocate a budget to each category to stay on track with your financial objectives.

As you make purchases, easily add them to the corresponding category within Budget Patrol. The extension will automatically deduct the purchase amount from the allocated budget for that category, providing real-time updates on your spending.

Stay informed about your overall financial health with a glance at the total allocated budget and remaining balance. Dive deeper into individual categories to monitor how much you've spent and how much budget is left, ensuring you stay within your financial limits.

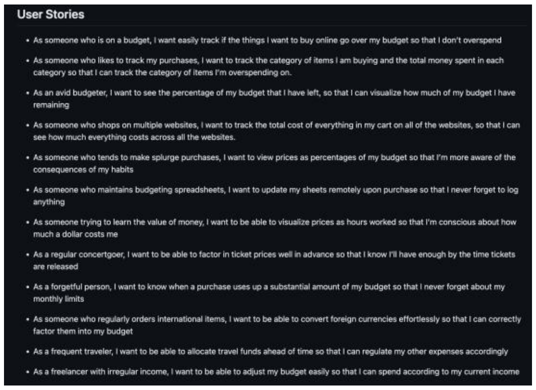
Budget Patrol is the ultimate tool for anyone looking to cultivate healthy spending habits and achieve their financial goals effortlessly. Take charge of your finances today with Budget Patrol – your faithful companion for smart budgeting right in your Chrome browser.

2 Design Process

2.1 Ideation

As college students ourselves, our product was fittingly inspired by the plight of enjoying online shopping despite not having a consistent income stream. Furthermore, we also observed that traditional budgeting methods like spreadsheets feel rather laboriouos, and who wants to deal with yet another chore when online shopping as a break from work? We came up with Budget Patrol as a way to give students and other budget-conscious online shoppers the discipline from these budgeting tools, while being much more pleasant to use.

2.2 User Stories



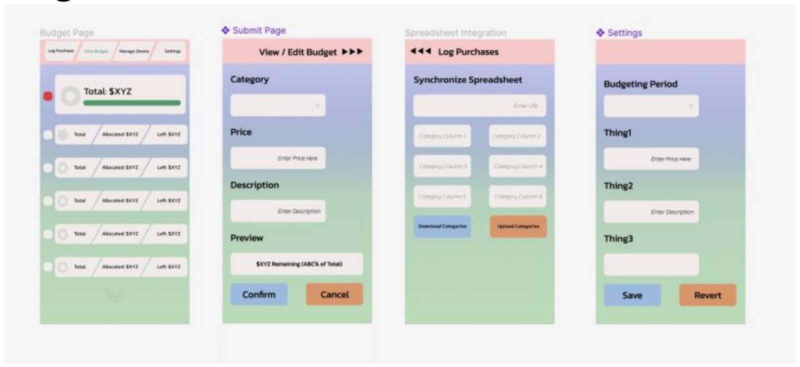
User Stories

- As someone who is on a budget, I want easily track if the things I want to buy online go over my budget so that I don't overspend
- As someone who likes to track my purchases, I want to track the category of items I am buying and the total money spent in each category so that I can track the category of items I'm overspending on.
- As an avid budgeter, I want to see the percentage of my budget that I have left, so that I can visualize how much of my budget I have remaining
- As someone who shops on multiple websites, I want to track the total cost of everything in my cart on all of the websites, so that I can see how much everything costs across all the websites.
- As someone who tends to make splurge purchases, I want to view prices as percentages of my budget so that I'm more aware of the consequences of my habits
- As someone who maintains budgeting spreadsheets, I want to update my sheets remotely upon purchase so that I never forget to log anything
- As someone trying to learn the value of money, I want to be able to visualize prices as hours worked so that I'm conscious about how much a dollar costs me
- As a regular concertgoer, I want to be able to factor in ticket prices well in advance so that I know I'll have enough by the time tickets are released
- As a forgetful person, I want to know when a purchase uses up a substantial amount of my budget so that I never forget about my monthly limits
- As someone who regularly orders international items, I want to be able to convert foreign currencies effortlessly so that I can correctly factor them into my budget
- As a frequent traveler, I want to be able to allocate travel funds ahead of time so that I can regulate my other expenses accordingly
- As a freelancer with irregular income, I want to be able to adjust my budget easily so that I can spend according to my current income

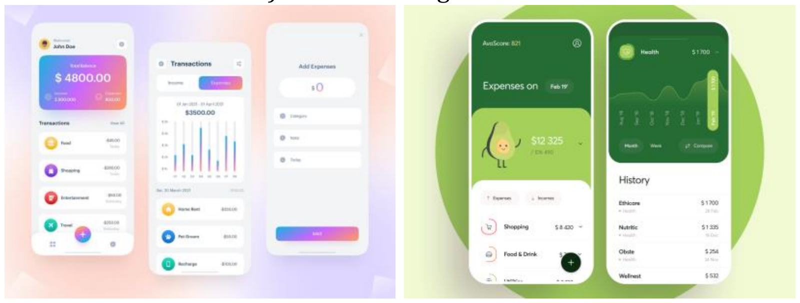
i) A selection of our User Stories

To come up with useful features, we wrote User Stories to better understand our target demographic and their needs. Naturally, the majority of our proposed features are geared towards those wanting to improve their fiscal responsibility, but we also imagined use cases for more specialized features like Google Sheets integration.

2.3 UI Design



ii) Our initial Figma draft

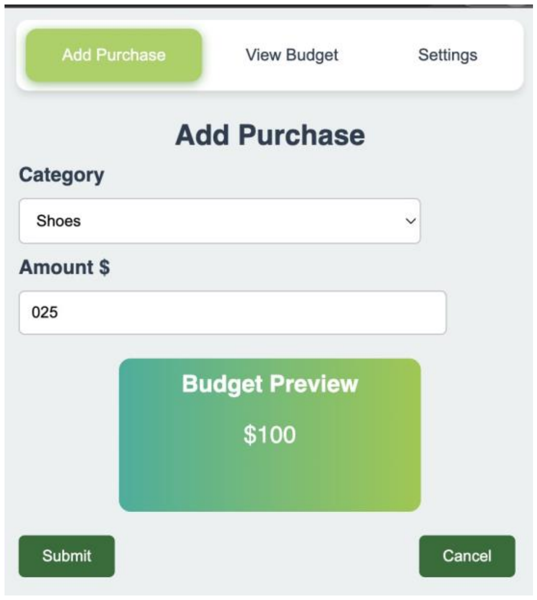


iii) Two UI mockups we used for inspiration

Having decided on a candidate feature set, we designed a Figma to better visualize the extension. Our proposed design is split into 4 tabs: View Budget, Add Purchase, Spreadsheet Synchronization, and Settings. Drawing inspiration from high-quality Budgeting App UIs found online, we focused on easy readability and a minimal aesthetic so that the user has an easy time navigating and using the extension.

3 Features

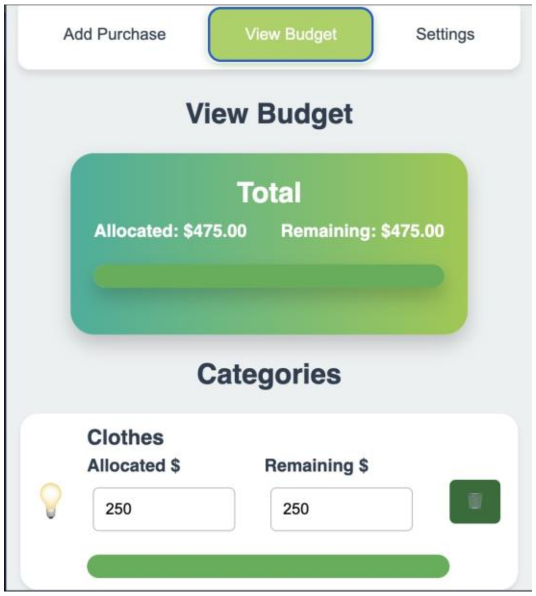
3.1 Add Purchase



i) A user logs a $25 shoe purchase, leaving $100 of their $125 shoe budget

Originally, we wanted the extension to automatically scrape prices from a website a user is on and provide a preview of what their budget would look like if they made this purchase. We realized that scraping prices is difficult due to different ways in which price is represented in a site's HTML and some sites block this processing. Instead of having the extension be limited to only working on site's where we could automatically scrape prices, we decided to drop the price scrapping feature and instead create an input field where a user can type in the price of the item they are interested in. We still implemented the budget preview feature by allowing the user to select the budget category that the item they are considering would fall into, type in the price of the item, and display how much would remain in their budget for this category if the purchase was made. This is a helpful feature to include in a budgeting tool because it allows the user to quickly check a purchase would take up a substantial amount of their budget (or even go over) and make an informed decision rather than making a purchase and then later checking its effect and realizing it was not a smart financial decision. On the Add Purchase tab we also have a submit button where a user can confirm a purchase and their budget is updated accordingly.

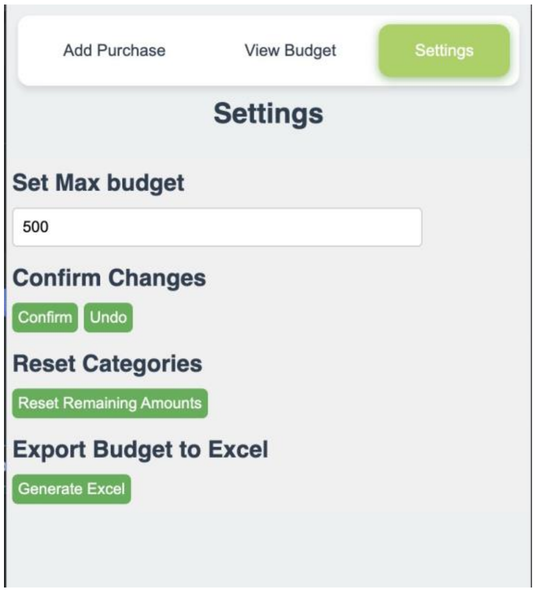
3.2 View Budget



ii) A user inspects the amounts allocated and remaining in their custom categories

Overall, the implementation of the View Budget tab followed our original plans and design. A particular feature we wanted to implement was allowing a user to add/delete budget categories and see a general overview how much they have spent in each category and overall. We decided to include this feature because it saves time for the user so that they don't have to click to another tab to see their budget breakdown/progress and instead can see it quickly on their current page to make an informed financial decision. At first, we weren't sure whether to but the add/delete category functionality in the Settings tab or in the View Budget tab, but we decided include these functionalities directly in the View Budget page so that a user could clearly see what categories already exist when adding a new category and make its more visible how to delete a category with the trashcan icon next to each one.

3.3 Settings



Settings

Set Max budget

500

Confirm Changes

Reset Categories

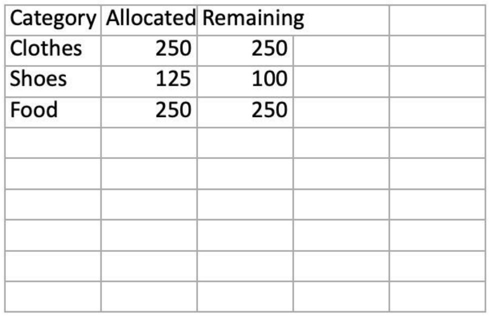
Export Budget to Excel

iii) A user sets the max budget per category. They can also replenish their budget and generate spreadsheets from here

Our initial plan for the Settings page was to allow the user to specify the maximum dollar amount a user can allocate for a single category, and a toggle for the refresh period-we hoped to let the user have their budget replenished every week, month, or year.

While we were able to implement the maximum budget field input using localStorage in SettingsTab and CategoryItem, we decided to not move forward with the refresh rate due to time constraints. Instead, we compromised on a manual "Reset Categories" button which allowed the user to manually replenish their category budgets. Again using localStorage, these changes are reflected on the ViewBudgetTab.

3.4 Excel Export



Category	Allocated	Remaining		
Clothes	250	250		
Shoes	125	100		
Food	250	250		

iv) A spreadsheet generated from the user's categories. This spreadsheet is up-to-date with the user's purchase history

We have added a method such that users can export their data into an excel file so that they can store it locally for whatever reason they want - to historically track/store their budgets, use in other applications, etc. We originally proposed using the Google Sheets API for this but ran into a few issues when connecting, mainly because of some authentication issues. It was much easier to just export as an excel file which is quite seamless.

4 Tools

1. **JavaScript/React:** Main scripting language used in the project We specifically used React scripts to help set up and simplify our react environment for the extension
2. **XLSX package:** Allows us to directly interact and produce xlsx files which helped us export the budget as an excel file.

5 Challenges

One challenge that we had was with the Google Sheets API. The API required OAuth which was tricky to set up for the chrome extension. To still give our extension some functionality with regards to spreadsheets, we made an export to excel function which allowed users to download their budget to an xlsx file so they can store it locally.

We also had to figure out how to pass variables between pages and save the user's categories + purchases between sessions. It turned out that the Chrome Local Storage API solved both problems, as we could use it to read in the most up-to-date value of variables, even those defined in other pages. This allowed us to do things like allow the user to set a maximum budget per category in the settings page, which is then enforced in the ViewBudgetTab.

6 Lessons + Takeaways

One lesson we learned during the build process was how project planning isn't exactly oneto-one with actual implementation. For example, we assumed certain parts of the building process would be simple when in reality they were a lot more complex than we expected. Examples of this include price scraping, integrating the google sheets API, and automatic budget refresh. Something we can take away from this is that it is important to have an understanding of what features may be complicated to implement and have a backup plan for if we can't include these features or allocate more time to their implementation during the planning process.

Also another lesson learned was coordination in our group - while Slack was useful for communicating updates and sending links and resources, our project management could have been enhanced by using GitHub projects. Sometimes, it was unclear whether a particular feature was in progress or not started and who was working on it. Similarly, a very important lesson we can take away is the importance of communication and supporting group members. We each became busier with other coursework demands throughout the semester at different times, but we did not do the best job of communicating to the group the times in which we had limited capacity and couldn't work on the project much and when we had more availability. This sometimes made it unclear what features would get done when and by whom and made tasks fall more on certain team members.

Regardless, our project creation experience had a ton of positives to take away. For example, we created a Figma diagram at the beginning to showcase how our end product should look and our chrome extension has held up quite closely to that diagram.