*Budget Watch*

Tyler Phenix

UMKC – School of Computing and Engineering

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Last User to make Changes** | **Date** | **Summary of Changes** |
| T. Phenix | 8/11/16 | Vision statement, product backlog, sprint plan |
| T. Phenix | 8/25/16 | Iteration 1 Spring Backlog, Review, Retrospective, and Velocity |
| T. Phenix | 10/2/16 | Iteration 2 Spring Backlog, Review, Retrospective, and Velocity |
| T. Phenix | 10/23/16 | Iteration 3 Spring Backlog, Review, Retrospective, and Velocity |
| T. Phenix | 11/6/16 | Iteration 4 Spring Backlog, Review, Retrospective, and Velocity |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table of Contents

[Vision Statement 4](#_Toc459581887)

[Requirements 5](#_Toc459581888)

[Step #1: Identify Categories of Users 5](#_Toc459581889)

[Step #2: Create Actor-Goal List 5](#_Toc459581890)

[Step #3: Identify User Stories 6](#_Toc459581891)

[Product Backlog 7](#_Toc459581892)

[[Step #4: Write Use Cases] 8](#_Toc459581893)

[Sprint #1 12](#_Toc459581894)

[Review 12](#_Toc459581895)

[Retrospective 13](#_Toc459581896)

[Design 14](#_Toc459581897)

[Coding Standards 15](#_Toc459581898)

# **Vision Statement**

I will build an app that allows the user to enter a budget for a shopping trip and allow the user to input the items as they go to make sure they stay on budget called “Budget Watch”. This will help with people who overspend during a shopping trip or anyone who wants to be stay organized while shopping. This is most important for parents and people who don’t have much money to spend (college students, low income families, etc.). Unlike other apps I have seen, Budget Watch gives helpful feedback to user to help stay ion budget.

My project will include a calculation feature, for matching the target budget amount against what the user has already spent. My project will also include a pop-up to let the user know if they are getting close to the budget or over budget. Also once you are done shopping, it will give you a rundown of how far over/under budget you were. My project will include a category to put each item in, either “need” or “want”. My project will not include a barcode scanner because prices vary based on different stores, store location, sales, etc. Instead the user will manually put in each item’s name and price (possibly a picture of the item also).

Essential features of my project include calculating the total price and matching it against the target budget and feedback to the user (whether they were on budget or not).

Desirable features include allowing the user to take a picture of the item and saving the shopping trip list for future reference.

# **Requirements**

## Step #1: Identify Categories of Users

**User**

**Parents** – Adults with a child or children who can use smart phone apps well.

**College students/recent graduates** – People who are older teens up to about age 30 who are saving money to pay for/back school expenses.

## Step #2: Create Actor-Goal List

|  |  |
| --- | --- |
| **Actor** | **Goal** |
| User | Input budget |
|  | Input items as shopping trip continues |
|  | Input name of item |
|  | Input price of item |
|  | Take picture of item if needed |
|  | Decide whether item is a need or a want |
|  | See results at the end of shopping trip |
|  | See total of entire shopping trip |
|  | See how much over/under budget shopping trip was |
|  | See total of wants and total of needs |
|  | See how much shopping trip would have been without wants |

## Step #3: Identify User Stories

As a parent, I want to spend as little as possible at the store so that I can pay other bills.

As a college student/recent graduate, I want to save money on groceries so that I can afford housing.

Product Backlog

The collection of stories makes up your project’s product backlog:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Story ID** | **Story** | **Story Points** | **Priority** | **Status** |
| S1 | Allow user to enter budget | 2 | 4 | Complete |
| S2 | Allow user to enter name of item and its price | 5 | 4 | Complete |
| S3 | Allow user to take picture of item | 7 | 2 |  |
| S4 | ~~Allow user to select if item is need or want~~  Add code for budget activity and calculations | ~~6~~  4 | ~~1~~  ~~4~~ | Complete |
| S5 | Allow user to get feedback during shopping trip based on total relative to budget | 6 | 4 | Complete |
| S6 | Allow user to see totals at the end of shopping trip | 4 | 4 | Complete |
| ~~S7~~ | ~~Allow user to see how much they could have saved if they had no “wants”~~ | ~~3~~ | ~~3~~ |  |

# **Sprint #1**

Sprint Backlog

Iteration 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Story ID** | **Story / Task** | **Estimated**  **Hours** | **Actual**  **Hours** |
| S1 | Design UI | 3 | 3 |
|  | Add code for storing input | 3 | 4 |
|  | Add about menu | 2 | 3 |
| S2 | Add features to layout | 3 | 3 |
|  | Add code for running total | 4 | 4 |
| S5 | Design pop-up menus | 5 |  |
| S6 | Add button for checkout | 3 |  |
|  | Add code for calculating totals | 4 |  |
| S7 | Add code for calculated new totals | 4 |  |
| S3 | Import picture library | 3 |  |
|  | Link picture to item | 3 |  |
| S4 | Add selection menu for each item | 3 |  |
|  | Add code for storing selection of each item | 4 |  |
|  |  |  |  |

Review

**Iteration 1:**

S1: Has actually taken about 10 hours and it is still not finished.

S2: Has taken about 7 hours as predicted but it is still not finished.

Retrospective

**Iteration 1:**

For this iteration, I planned on finishing two stories. I did not completely finish neither. The main reasons for this is I misinterpreted how long each task would take. My “about menu” was supposed to be fairly quick but I could not get it to work. Also for Sprint 2 I could not get the numbers to be parsed in correctly. The amount of time each task takes will have to be recalculated in the future to better manage my time on each sprint.

For Iteration 2 I plan on finishing Sprint 1 and 2 and also 5.

**Project velocity: 0**

# **Sprint #2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Story ID** | **Story / Task** | **Estimated**  **Hours** | **Actual**  **Hours** |
| S1 | Design UI | 3 | 3 |
|  | Add code for storing input | 3 | 4 |
|  | Add about menu | 4 | 6 |
| S2 | Add features to layout | 3 | 4 |
| S5 | Design pop-up menus | 5 | 5(so far) |
| S6 | Add button for checkout | 6 |  |
|  | Add code for calculating totals | 10 | 5(so far) |
| ~~S7~~ | ~~Add code for calculated new totals~~ | ~~4~~ |  |
| S3 | Import picture library | 3 |  |
|  | Link picture to item | 3 |  |
| ~~S4~~ | ~~Add selection menu for each item~~ | ~~3~~ |  |
|  | ~~Add code for storing selection of each item~~ | ~~4~~ |  |
|  |  |  |  |

**Iteration 2:**

S1 and S2 are complete and parts of 5 and 6 are started.

Retrospective

**Iteration 2:**

This iteration was much more successful than my last one. I was able to recalculate the times and story points after realizing what I got myself into. This has made things a little better since I can plan my sprints out betters. I finished my “about menu”, I also got most the “running total” working but realized it is a much larger task so I broke it up some. This is because I am working with a “string list” and I need to pars the prices in as integers which is much easier said than done. So I did end up finishing sprint 1 and 2and started deeply working on 5 and the “totals” part of 6.

So that I do not plan too much for my plate, I plan on finishing up the “totals” component of both sprint 5 and 6.

**Project velocity: 7**

# **Sprint #3**

|  |  |  |  |
| --- | --- | --- | --- |
| **Story ID** | **Story / Task** | **Estimated**  **Hours** | **Actual**  **Hours** |
| S1 | Design UI | 3 | 3 |
|  | Add code for storing input | 3 | 4 |
|  | Add about menu | 4 | 6 |
| S2 | Add features to layout | 3 | 4 |
| S5 | Design pop-up menus | 5 | 8 |
| S6 | Add button for checkout | 6 |  |
|  | Add code for calculating totals | 10 | 8 |
| S3 | Import picture library | 3 |  |
|  | Link picture to item | 3 |  |

**Iteration 3:**

S6 is almost complete. I got the total working but due to poor design choice, which so happens to be the topic of this iteration, I am unable to get the budget working properly at this time.

Retrospective

**Iteration 3:**

Since I didn’t finish S6 my velocity is zero but I got the most done that I have in any iteration so far. The next step will be redesigning my budget activity so that I can call the updateBudget function, which is located within the main activity, from within it. The design my app is currently in does not allow this since updateBudget is not static. It cannot be static because it uses “findViewById” which does not allow for static functions. My next iteration should be interesting. I plan on finishing S6 for my next iteration. Once I get the budget working, I will be able to finish S6, which maybe should have been broken up into smaller stories.

**Project velocity: 0**

# **Sprint #4**

|  |  |  |  |
| --- | --- | --- | --- |
| **Story ID** | **Story / Task** | **Estimated**  **Hours** | **Actual**  **Hours** |
| S1 | Design UI | 3 | 3 |
|  | Add code for storing input | 3 | 4 |
|  | Add about menu | 4 | 6 |
| S2 | Add features to layout | 3 | 4 |
| S5 | Design pop-up menus | 5 | 8 |
| S6 | Add button for checkout | 6 | 5 |
|  | Add code for calculating totals | 10 | 8 |
| S4 | Add code for budget activity and calculations | 10 | 10 |
| S3 | Import picture library | 3 |  |
|  | Link picture to item | 3 |  |

**Iteration 4:**

I completed S6 and added a new story for the budget(S4) which was also completed. I also added two exceptions and an assertion.

Retrospective

**Iteration 4:**

I finished adding code for budget which I separated into its own story(S4). The budget now transfers properly. Since our topic this iteration is error handling, I’ve added both two exceptions and one assertion. In case the user leaves the budget blank, I’ve added an exception to allow them to continue without a budget or give them another chance to enter their budget. I also added an exception in case the user tries to leave the item and/or price empty before adding to the list. It now gives them a message telling them what they need to do if they have no added one or the other. The assertion in case my switch statement for changing activities fails.

**Project velocity: 8**

# **Design**

[What is the overall structure of the solution? What are the major modules of code? What are the dynamics of communication between these modules? The most common way of depicting this information is with static and dynamic models augmented with short narrative descriptions of design.]

# **Coding Standards**

Coding standards improve readability. They make it easier to understand code written by others. Good coding standards also improve reliability.