Team: Imposters 4156 T1

Name	Yumeng Luo	Kevin Hwang	Sang Jun Chun	Shengtan Mao
UNI	yl4655	jch2169	sc4658	sm4954

Part 1:

1.1 Description:

Our tool is called “saving and spending’s tracker” which will keep track of the user's spending and savings. It will record user’s each transaction and prompts an alternative same category purchase that's cheaper in price but can fulfill the same need. (For example, one user is planning to spend $5 on a cup of Starbucks coffee, the tool will propose an alternative - a cup of Mcdonald’s coffee for $2) If the user did choose the alternative, the tool will calculate the difference in price as saving and add the savings to account balance. (In the previous example, the tool will record a spending of $2 and a saving of $3 if the user chose the latter) At the end of each week, the tool will show balance accumulated so far and users can choose to reward themselves with the money saved or spend it in other ways, or even donation. (The user spent a total of $1000 and saved $100 at the end of the week and chose to reward himself with a gaming mouse or donate to a charity group of choosing.)

1.2 Users for our tool include:

1. Users who are on a tight budget and want to know cheaper alternatives to make ends meet
2. People who're saving money to purchase particular items or occasional indulgence
3. Advanced user who wants to customize which purchases to save from
4. People who wish to give back to charity in small ways (with money saved from daily expenses)
5. Sellers who're promoting their products that’s a cheaper alternative to an existing one

1.3 Saving data:

Our tool will require each user to register and login before using and will be forced to timeout after 20 minutes of inactivity. The account balance, spending and saving’s history of each individual user will be saved persistently using a database and will only be shown to the account owner after login.

1.4 Demonstration:

Our current plan for demonstration is through the command line console: User will perform registration, login, record spending, choose alternative, check balance, spend balance and logout in command line. This may not be ideal for visualization, therefore we will try to add a website GUI if time permits.

1.5 API

For API usage, we are planning to use “Charity Search” to search for charity the user can donate their savings to, “Plaid” to connect to the user's financial accounts, “Walmart Open API”,”Best Buy” and”Wegman” to generate alternatives.

Part 2:

Our MVP should be able to keep track of spending and prompt cheaper alternatives; keep track of savings and reach user set goals; customize alternative option's categories. The detailed user stories are as following**:**

USId1. As **a user with a tight monthly budget**, I want to **know and choose cheaper alternatives for daily expenses** so that **I can stay under budget and save money**. My condition of satisfactions are:

a. I can set a budget I want to spend under

b. The tool correctly records my spendings

c. The tool proposes a cheaper option that also satisfy my need

d. The tool correctly records my savings

e. Following the tool's suggestions, I can spend less than my budget permits

USId2. As **an everyday user looking for occasional indulgence**, I want to **save on daily expenses** so that **I can purchase more expensive items**. My conditions of satisfactions are:

a. The tool correctly records my savings

b. I can set a goal to save up to

c. Following the tool's suggestions, I can save more than my goal

USId3. As **a user looking for cheaper alternative options**, I want to **specify what categories of purchases are extraneous to me** so that **I can save more in such categories**. My conditions of satisfaction are:

a. I can customize the items I feel are extraneous

b. The tool proposes the cheapest alternatives in these categories

Aside from the above 3 user stories, we propose the 2 following optional user stories that should be completed if time permits**:**

USId4. As **a user who wants to give back in small ways**, I want to **know and choose what charity organizations are seeking donations** so that **I can donate some of my savings from daily expenses to a charity of my choosing**. My conditions of satisfaction are

a. The tool shows descriptions of potential charities I can donate to

b. I can specify which charity I'm donating to and what amount I am donating

c. The tool returns confirmation that the charity has received my donations

d. The tool makes my donation anonymously

USId5. As **a consumer product company focusing on cheaper goods**, I want to **have my products shown as cheaper alternatives for similar purchases by tool's users**, so that I **can effectively market my products to the right customer base.** My conditions of satisfaction are

a. My product is proposed to users as an alternative to a more expensive product

b. Consumers should be able to purchase from the platform (app, website etc)

c. The tool provides report on who / how many of my products are being sold

Part 3:

USId1. Common Case

a. The tester sets a budget amount.

b. The tester purchases multiple products and verifies that the tool correctly records spendings based on the products chosen.

c. The tester requests multiple alternative recommendations and verify those products fulfills the purpose of the original product and are cheaper in price.

d. Tester makes choices for alternative options for multiple products and verifies that the tool correctly records the savings based on the product chosen and the original product.

e. Once the budget amount has been reached, the tool should warn the tester budget reached and proposed alternatives should be “do not make this purchase”.

f. The tool will pass this test if the final monthly spending is less than the set budget, otherwise the test fails

USId1. Special Case

a. The tester tries to set the budget as a negative number. We expect an error that the user is trying to set an invalid value.

b. If the tester does not set the budget and proceeds, the tool will warn the tester the budget is not set.

USId2. Common Case

a. The tester sets a savings goal. Testers should not be able to set the goal to negative numbers.

b. The tester records purchases for multiple products and makes choices for alternative options and verifies that the tool correctly records the savings based on the product chosen and the original product.

c. Once the savings amount has been reached, the tool should notify the tester that the goal met.

d. The tool will pass this test if the final monthly saving is more than the set goal, otherwise the test fails

USId2. Special Case

a. The tester tries to set the goal as a negative number. We expect an error that the user is trying to set an invalid value.

b. If the tester does not set the goal and proceeds, the tool will warn the tester the goal is not set.

USId3. Common Case

a. The tester unchecks all categories as “extraneous”

b. The tester makes purchases in multiple categories and records the price for each alternative options

c. The tester checks all categories as extraneous

d. The tester makes the same purchases as before

e. The test passes if the price of the alternative option is lower than the recorded price before checking as “extraneous”, otherwise the test fails

USId3. Special Case

a. The tester makes a purchase with a negative number as price.
b. The test passes if the tool warns the tester the price is negative and does not record this purchase or propose an alternative. Otherwise the test fails.

Part 4:

For our backend facilities, we plan to use the following:

- Eclipse for our IDE (using Java JDK 11)
- Maven for our build tool
- SQLite for our database

For testing, we plan to use the following:

- JUnit for unit testing
- Emma for code coverage

For other tools, we plan to use the following:

- CheckStyle for our style checker
- Spotbugs for our bug finder