Part0

We changed our planned programming language of backend, which is Python, to Java. Because after discussion, we think Java is more suitable for backend programming, and we have learnt a lot of Java tools like Junit and JDBC in former lectures.

Part1

1.1. What will your project do?

Our project is to implement a web application that mainly acts as an official website for iDrop. And iDrop is a fictitious company that focuses on getting profit from salvage of waste material. Hence, our website will show the users what our services are about and provide services for them.

1.2. Who or what will be its users?

Our users are other some other companies that want to cooperate with us, and normal citizens. All the users can visit our website as guests, but if they want to be served or develop business cooperation, they should register a unique account with password.

1.3. demoable

We will be able to show the structure of our website. And the sign up and login in process of the web. We will also try to implement some interaction design of our service. The details are still being discussed.

1.4. What kind of data do you plan to store?

We plan to store users' data, such as username, password, the service they choose and the billing account.

1.5. What API do you plan to use and what will you use it for?

We plan to use Calendar, Business and some Google API. There might be more during implementation.

Part 2

2.1 As a user, I want to cash the money from my account so that I can get my rewards of recycling the waste.

My condition of satisfaction is that the balance on my account is more than 0.

2.2 As a user, I want to accumulate money on my account as I recycle more and more waste so that I can make a one-off withdraw.

My condition of satisfaction is that every time I recycle the waste, I use the same account and take the money.

2.3 As a manager, I want to assign different rewarding price to the waste so that they have different values.

My condition of satisfaction is that the waste users recycle is not beyond the list.

Part 3

3.1 For the first user story, a user logs in and he has recycled the waste.
Common case: Either our manager or a device will confirm and classify the waste, and give a signal to our system, i.e. input. The output is the corresponding increment on the user's account, which passes the test.
Special case: The waste is beyond our recycling list, i.e. input. The output is an error message instead of the increment on the user's account.

3.2 For the second user story, a user has had money on his account.
Common case: The user logs out the system and then logs in, which are two inputs. Then he checks his account, which is another input. If money in the account is consistent as before, the test passes.
Common case: The user sends a request to withdraw an amount of money, i.e. input. The output is a corresponding reduction on his account, and an increment on another account like a debit card.
Special case: The user sends a request to withdraw an amount of money, which is beyond the balance, i.e. input. The output is an error message.

3.3 For the third user story, only pre-set kinds of wastes are acceptable.
Common case: The inputs are the kind and the amount of the waste. The output is the corresponding rewards.
Special case: If the waste is beyond our recycling list, the output is an error message.

Part 4

We plan to use JavaSE-11 as the compiler, Eclipse as the IDE, Maven as the build tool, CheckStyle as the style checker, JUnit as the unit test tool, Emma as the coverage tracking tool, Spotbugs as the bug finder and SQLite as the persistent data store.