Differentiation

Yanjia Zhao, yz476
Jiaran Zhou, jz270

Beside the basic functions in the requirement document, we also add some features to differentiate our UPS service, both in Functionality and performance.

1. Functionality

We provided several extra functions so that our UPS website works more like a real UPS!

1.1 Change destination from Amazon side

Beside change destination from UPS side, we can also change destination from Amazon side. We designed this functionality because sometimes users may want to change destination without login to their UPS account. We realized this functionality by adding another two messages in our protocol.

In Amazon to UPS:

```
message request_change_destination {
        required int64 package_id = 1;
        required int32 new_destination_x = 2;
        required int32 new_destination_y = 3;
}

message AUCommands {
        optional request_get_truck get_truck = 1;
        optional request_init_delivery init_delivery = 2;
        optional request_change_destination change_destination = 3;
        optional bool disconnect = 4;
}
```

In UPS to Amazon:

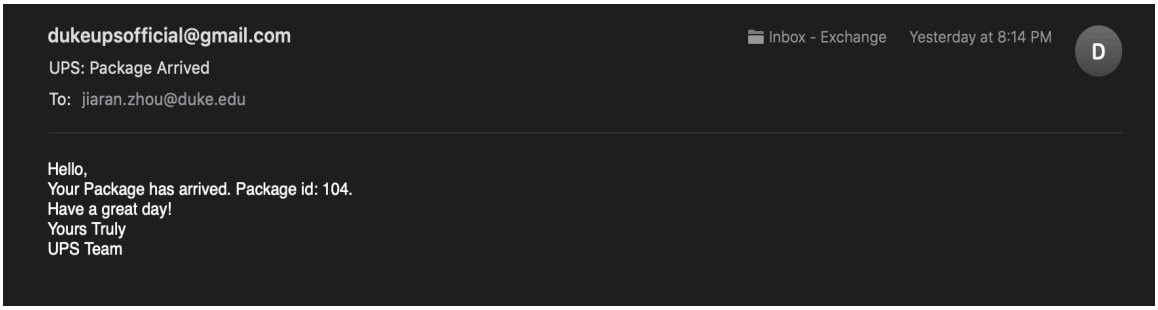
```
message response_destination_changed {
        required int32 new_destination_x = 1;
        required int32 new_destination_y = 2;
        required int64 package_id = 3;
        required bool success = 4;
}

message UACommands {
        optional response_truck_arrived truck_arrived = 1;
        optional response_package_delivered package_delivered = 2;
        optional response_destination_changed destination_changed = 3;
        optional bool disconnect = 4;
        optional int64 world_id = 5;
}
```

When a user clicks change destination on Amazon website, Amazon server will send us an AUCommands which contains a request_change_destination message. Our UPS server will search our database to see if the package is out for delivery or not. If not, our server will change its destination in database before Godeliver UCommand is sent. Then it will respond to Amazon server with a UACommand, which contains a destination_changed message. The success field will be True. Otherwise, it will not change the database and the success field in message will be False. Amazon side will also handle our message. In this way, both Amazon and UPS side will see the result of destination change.

1.2Send email

We have registered a gmail account to send emails to users. We set up our SMTP authentication in server code. Each time a package is delivered, our server will send an email to user as follow:



By providing the same email address for UPS account and Amazon account, a user can receive both emails when a package is delivered and arrived.

2. Performance

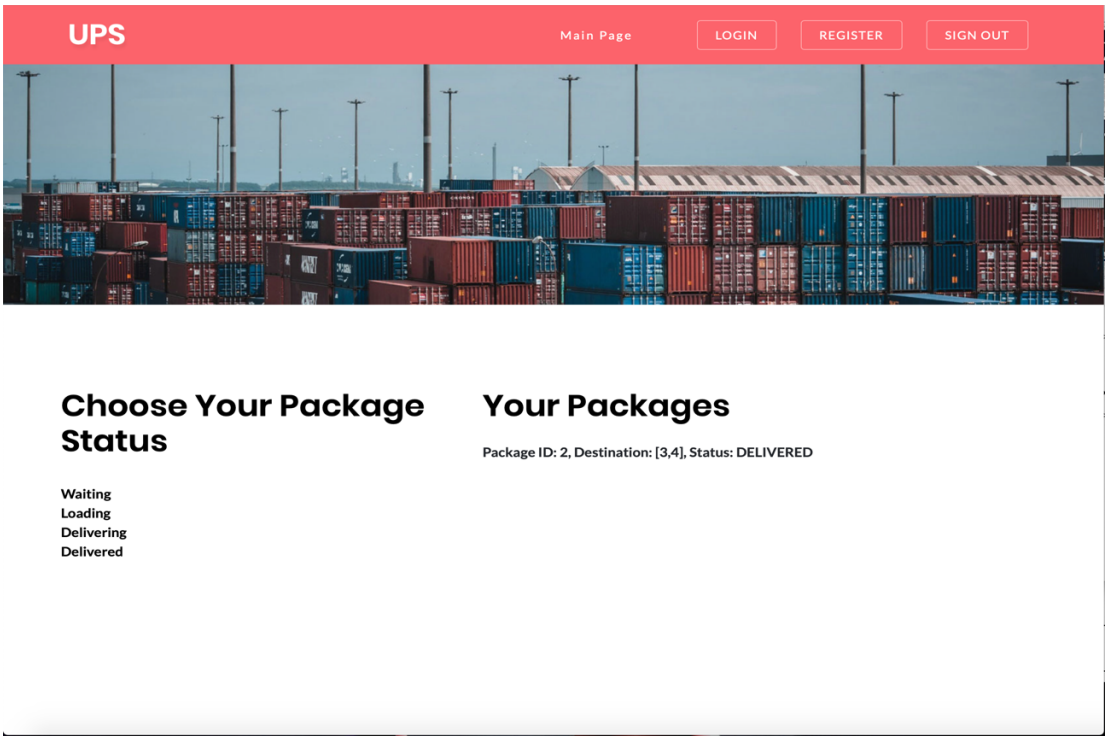
2.1Security

Our website has verification that users are not allowed to register if they have the same username or email address.

If a user provides a wrong UPS account on the Amazon side, our UPS server will first check it and do not associate any UPS account with this order. It can only be searched by its package id.

2.2Classify Packages by Status

In the user's home page, he can see all his packages. What's more, we add more status to the packages, so they are classified as WAITING, LOADING, DELIVERING and DELIVERED. He can click on those tags, so that he can only see packages of one of those status. He can only click on packages which are waiting or loading to change the destination. The classification of different kind of packages is as follow.



2.3Package History

The user can not only see the current status of his package, he can also see the history of each package of his. For example, if the package status is DELIVERED, it will also show the time when the package is created, loading, delivering and delivered, the screenshot is as follow.