**Requirements Definition**

**Introduction**

This project aims to build a system that allows customers to post yard work-related jobs and for workers to accept those jobs.

The system will allow customers to post a job with its listing details. The system will allow workers to accept jobs that they want. The system will allow customers to pick which worker they want to work on the job. When a worker notifies the system that they have completed the job the system will transfer the price listed by the customer from the customers account to the workers account while taking a 10% cut. If the customer and worker continue to work with each other and decide to stay in contact over the phone rather than through posting job listings on the web app, then the customer can continue to pay the worker directly through the web app where the system will take a 5% cut on said scenarios.

**Users and Their Goals**

For further information on users and their goals, view the use case diagrams in /docs/

Owner:

Aims to make money by providing a service through which customers may post yard work-related jobs to a site and have them be accepted.

Customer:

Aims to pay for a worker to perform labor on their behalf around their property.

Worker:

Aims to earn money by accepting jobs from customers and performing their tasks.

**Functional Requirements**

1. User Authentication
   1. The system will require all users to log in in order to view their account and associated info
      1. Upon logging in for the first time, users will provide a valid email address and password for future login.
         1. Additionally, they must specify or provide the following:

* Whether they intend to use the service primarily as a worker or as a customer.
* A valid payment method to use the service.
* A neighborhood or street address.
* Optionally, a phone number.
* A username
  + 1. For future login attempts, the system will require users to provide their email address and password which the system will then either validate and grant access to the account, or invalidate and allow the user another chance to log in.
       1. Should the attempt to login fail, the system will make no indication of whether the email address was incorrect or the password was incorrect, simply that one or both was incorrect. This is for security.
  1. The system will prevent the same email address from being used in multiple accounts

1. User Access
   1. The service owner may have owner, worker, and customer access rights; however, other users may only have a combination of customer and worker access rights.
      1. The user with owner rights shall have access to all owner features. See FR #4
      2. Users with customer rights shall have access to all customer features. See FR #5
      3. Users with worker rights shall have access to all worker features. See FR #6
      4. Users who aren’t the owner may modify their status as workers and/or customers at any time.
2. User Profile Features
   1. The system will allow all authenticated and logged-in users to modify their own email address.
      1. The system will prevent any user from modifying any other account’s email address.
   2. The system will allow all authenticated and logged-in users to modify their own password.
      1. The system will prevent any user from modifying any other account’s password.
   3. The system will allow users to view their account balance
   4. The system will allow those with owner and customer access to transfer funds from their payment source to their account balance.
      1. For the security of their funds and to prevent laundering, workers without customer access will not be allowed to transfer funds to their account.
   5. The system will allow all users to transfer funds from their account balance to their payment source.
   6. The system will allow users to change or remove their payment method.
3. Customer Features
   1. All users who elect to use the service as a customer shall have access to all customer features.
   2. The system will allow all users with customer access the opportunity to create a new job listing.
      1. Upon listing a job, the customer must specify their general location (Zip code), estimated time to complete the job, a desired day of the week, and a fixed amount they intend to pay for completion of the job.
   3. Customers will be able to see all of their listings and edit any of them prior to them being accepted by a worker.
      1. The user may deactivate any listing at any time prior to it being accepted by a worker.
   4. Customers may search for any worker’s username and send them money at any time with a 5% fee being deducted from the amount and sent to the owner’s balance.
   5. When a worker accepts a job and the customer accepts the worker, the customer may view that worker’s contact information to get in touch.
   6. After a worker claims to have finished a job, the advertised listing price is deducted from the owner’s account and transferred to the worker’s balance with a 10% fee being deducted from the amount and sent to the owner’s account balance. Additionally, the customer will receive a notification prompting them to elect:
      1. To tip the worker.
      2. To assign a star rating to the worker.
      3. To blacklist the worker.
   7. After a job is completed, the customer has 12 hours to contest the claim on the grounds of incompletion or crime. Should they prove their claim, the price will be deducted from the worker’s balance and re-added to their own account.
   8. If after 24 hours, an active listing hasn’t been accepted, the system will send a notification to the user encouraging them to modify their listing.
4. Worker features
   1. All users who elect to use the service as a worker shall have access to all worker features.
   2. The system will allow all workers to view a list of all currently active job listings
      1. Listings will display the category of work and a price for the job (minus the 10% imposed by the owner).
   3. Workers may select a job listing from the list to view additional information.
      1. The general location (zip code) of the job, the time range for the job, and the customer’s username and rating are displayed after selecting the listing.
      2. A worker may accept a job from within this expanded view.
   4. When a worker accepts a job and the customer accepts the worker, the worker may view that customer’s contact information to get in touch.
   5. After completing a job, a worker can indicate they have completed the job. At this point, the worker will receive the listing price they saw and accepted (which is 90% of what the customer posted) and a notification allowing them to:
      1. Assign a star rating to the customer
      2. Blacklist the customer.
5. Owner features
   1. The owner will automatically have access to all customer and worker features.
   2. The owner will receive 10% of every completed job, as well as 5% of every other user-to-user balance transfer (minus tips).
   3. The owner may blacklist any user from the service should the user prove unsatisfactory.
   4. The owner may view usage statistics for the system.

**Non-Functional Requirements**

1. The system will include both a site and an Android app
2. The system must be deployable
   1. Can be hosted by Heroku
3. The system will use the Git version control system.
4. The system will use a database
   1. The database will store the following about each user:

* Email address
* A username
* Hashed password
* Account type (Owner, Worker, or Customer)
* Payment info
* A phone number.
* A zip code
* Any listings they have created
* Any listings they have accepted
* Blacklist for other users
* Star rating (out of 5)

**Future Features**

1. A chat function
2. The user can edit or delete a listing after a worker accepts it and the worker will be notified
3. Should the user delete all payment methods from their account and display a zero balance in their account, their customer and/or worker access rights will become dormant, but their account will remain active.

**Glossary**

*System* - This service: Money LAWNdering and its accompanying website and app.

*User* - Any person who elects to use the system

*Customer* - A user who lists and pays for a service to be completed

*Worker* - A user who accepts job listings and gets paid for their work

*Owner* - A user who manages the system and receives a fee for all transactions