Jonathon Zempel
Capstone CS 401 Fall 2022

Rentable: The Item Rental Application

Contents

Proposal:

General Description:

This project is a mobile application that allows users to either lease, or rent tools and equipment. To lease an item, the user uploads a picture and an optional description, then sets the price per time period. Renters can then search by specific item, keyword, or location to find a tool they need. The renter can reach out to the leaser via built in messaging, which will help the two sides communicate while keeping as much privacy as both users feel comfortable.

The application will need to store users credit card information, to ensure payment; though cash will be the recommended form of payment. The app will help keep track of how long the item has been rented for, displaying different count times for leaser and renter.

Object/Purpose:

The purpose for this application is to connect people who need certain tools, but do not want to have to purchase them. Most hardware stores have rental programs, but are often overpriced and/or do not offer a wide enough selection. This service will also offer users to put their own tools up, allowing them to make some extra money.

Problem software will solve:

Instead of having to buy a specific tool that will only be used once, a user can just rent one for how ever long their project calls for it. Many people have tools and equipment that just sit around, only occasionally using them. This platform allows them to earn some extra money off those items.

Intended Audience:

This application is intended to be used by everyone. It may be a good idea to make it for only 18+, but the need to input a credit card will typically act as a deterrent for minors. The use is not inherently restrictive in age, so unnecessary for that restriction.
The typical user will be someone who takes on occasional DIY projects, only needing a piece of equipment for a single project or short term. It will also be used by those who need more expensive items, and don't have the funds or budget for everything they need.

Ethical Concerns:

Users could easily take advantage of "choose your own rate" ability of the application, over pricing. They could also rent out broken items, and claim that the renter was the one that broke them. On the same note, the renter could claim they were rented faulty items and refuse to pay, even after using them. The application could also become a platform for illicit items. There is no censoring of what type of item is added, so essentially anything could be put up.

Another issue which could come up, is a user renting an item, then putting it up themselves for a higher price; essentially subleasing it out.

Intended Platforms(s), Software Tools, Languages:

This is a mobile app, having cross functionality between AndroidOS and iOS. The creation will be through Android studio leveraging its UI layout building, and written mainly in java or Kotlin. It will use Azure/cloud to handle the back end, that the applications will use to connect through. It will use a SQL database to store all of the data, and firebase to store and verify login.

Key Features:

The application will offer users the option to either lend out or rent tools or small equipment. The user will be able to search for specific items, as well as refine by distance and price (or any combination of).

When putting items up for lease, the user will be able to present a picture and create their own price/term as well as a short description of the item.

There is a chat/messaging interface where the two parties can interact to allow for as much privacy as they wish: names or numbers do not need to be exchanged until terms are agreed upon.

When signing up, there will be terms presented that the user must agree to in order to proceed.

Renters will have to present a credit card for a deposit, which will be held by a 3rd party.

User login will require a password and all information will be stored on a secure database. Only one account will be allowed per email, and application will not be able to be used unless a user has an account.

The program will be inherently user friendly. Offering drop down menus and search capabilities. It will be visually pleasing and ordered in a way that makes sense. Users will be able to navigate the application with no prior knowledge. using elements from preexisting user interfaces and using an intuitive user flow.

There will be a built in timer that will be displayed for each item, both when renting and when leasing that will keep track of the amount of time it has been rented

Introduction:

This document contains the requirements needed to create project. It is broken into 4 sections: user and system requirements, research and development, UI design sketches, and UML diagrams.

1. User requirements:
    - 1.1. This product will connect individuals who either: Need to rent a tool or piece of equipment or have tools or equipment that they would like to rent out to people
    - 1.2. The users of this product will have the option to sign up as a renter, a leaser, or be able to do both on the same account
    - 1.3. When signing up for a new account
        - 1.3.1. Users will be presented with a page for them to fill out
            - 1.3.1.1. Will require a name, address, email
            - 1.3.1.2. Users will be required to accept terms of use
            - 1.3.1.3. If user is signing up as a Renter or later choosing to open up the ability to rent
                - 1.3.1.3.1. Will be required to input a valid credit card
        - 1.3.2. Renter:
            - 1.3.2.1. Search the available database of tools that other users have put up for rent
                - 1.3.2.1.1. Narrow search by distance and tool
                - 1.3.2.1.2. Send either a prewritten query or compose their own
                - 1.3.2.1.3. Purchase added insurance
        - 1.3.3. Leaser:
            - 1.3.3.1. Post items up for rent
                - 1.3.3.1.1. Add picture and choose their own rental rate
                - 1.3.3.1.2. When contacted can accept or reject request
    - 1.4. The users will be able to communicate between each other
    - 1.5. When opening the application, the users will be met with a login screen
        - 1.5.1. Will use email and a chosen password to login
            - 1.5.1.1. If cannot remember password, will have an option to reset
            - 1.5.1.2. If user has not created an account, there will be an option to sign up
        - 1.5.2. Once the user is logged in, it will open into their dashboard
            - 1.5.2.1. Will list any items up for lease, out on lease, or rented
            - 1.5.2.2. Will also include the amount owed per item rented
            - 1.5.2.3. Will be able to navigate easily away from this page via a menu
                - 1.5.2.3.1. Have options to search for new items, put more up for rent, view what is already put up for rent, and what items of theirs are being rented

2. System requirements:
   - 2.1. Functional Requirements
     - 2.1.1. The application will always validate any type of input and not allow anything other than expected inputs
       - 2.1.1.1. Will produce error messages if not a valid input
     - 2.1.2. Create new user
       - 2.1.2.1. Values: name, email, address, credit card, password/confirm password
     - 2.1.3. Log in
       - 2.1.3.1. There will be prompted fields for an email and password
       - 2.1.3.2. If the info input is found to be valid
         - 2.1.3.2.1. Logs in and allows access to dashboard
       - 2.1.3.3. If one or the other is found invalid
         - 2.1.3.3.1. Print out a try again message
         - 2.1.3.3.2. Reset fields and allow user to try again
       - 2.1.3.4. If user forgot password
         - 2.1.3.4.1. Prompt for an email
         - 2.1.3.4.2. Send password reset
       - 2.1.3.5. Reset password
         - 2.1.3.5.1. Prompt user to input a new password and then re enter the same password to confirm it
         - 2.1.3.5.2. Check to make sure is a valid password, otherwise print error and prompt to try again
           - 2.1.3.5.2.1. If password is valid, update database with new password
     - 2.1.4. Put item for rent
       - 2.1.4.1. Values: tool type, tool manufacturer, picture, description, rental rate
       - 2.1.4.2. Must include a picture and a price
       - 2.1.4.3. Must be then put onto a database to be accessible
     - 2.1.5. Remove tool
       - 2.1.5.1. Allow leaser to remove a tool that has been put up for rent
         - 2.1.5.1.1. Only allow this if the tool is not currently being rented
         - 2.1.5.1.2. Update database with removal
           - 2.1.5.1.2.1. Possibly keep data to allow user to easily relist
     - 2.1.6. Rent tool
       - 2.1.6.1. Needs to access the database
         - 2.1.6.1.1. Mark tool as rented
       - 2.1.6.2. Open up messaging between renter and leaser
       - 2.1.6.3. Prompt for insurance
         - 2.1.6.3.1. If the user wants to purchase insurance on the item
           - 2.1.6.3.1.1. Ask how much – charge credit card
             - 2.1.6.3.1.1.1. No refund for optional insurance
       - 2.1.6.4. Set rental term and rate

- 2.1.6.4.1. Leaser sets a price rate, and renter either accepts it or can haggle going back and forth until an agreed upon price
- 2.1.6.5. Decide if term is fixed or open
  - 2.1.6.5.1. Start rental timer at pickup
    - 2.1.6.5.1.1. Calculate total running cost if open ended rental
- 2.1.7. Message
  - 2.1.7.1. Take string inputs from either user and output them to the other
  - 2.1.7.2. Store messages on database
- 2.1.8. Search
  - 2.1.8.1. Need to take search parameters
    - 2.1.8.1.1. Location, tool type, tool name
  - 2.1.8.2. Query database with parameters
  - 2.1.8.3. Return matching items
    - 2.1.8.3.1. Name, picture, price, owner information (hidden during search)
- 2.1.9. Access database
  - 2.1.9.1. Need to be able to input new values into the database
    - 2.1.9.1.1. Input: picture, description, price, location, owner info
  - 2.1.9.2. Need to be able to make queries and return data
    - 2.1.9.2.1. Return: picture, description, price, location, owner info
- 2.1.10. Charge credit card
  - 2.1.10.1. Send outgoing credit card transactions
- 2.1.11. Rental timer
  - 2.1.11.1. Start either a count up or count down timer
  - 2.1.11.2. Send push notification when rental time is coming to an end
    - 2.1.11.2.1. If over time, start a new timer
  - 2.1.11.3. If a count up timer, also have a cost calculator of a running total
    - 2.1.11.3.1. If over time, start running cost total
  - 2.1.11.4. After a set amount of time after agreed upon term, open dispute of no return
- 2.1.12. Dispute of no return
  - 2.1.12.1. Leaser can open a dispute if item is not returned by agreed upon time plus set amount of extra time
    - 2.1.12.1.1. If claim is true charge tool cost plus, no return fee

2.2. Non-functional Requirements

- 2.2.1. Security
  - 2.2.1.1. Users information must be private, stored on a central database
  - 2.2.1.2. No user should have access to other users information
  - 2.2.1.3. Passwords need to be stored in a separate area from other data
  - 2.2.1.4. Credit cards need to be securely stored separate from other data
  - 2.2.1.5. After each session, users need to be automatically logged out
- 2.2.2. Memory
  - 2.2.2.1. The database that store the information needs to be able to scale with the inclusion of new items being put up for rent, and for new users signing up

    2.2.2.2. Minimal memory should be required for each user to install and use the application

2.2.3. Reliability

    2.2.3.1. The system shall not have any downtime during normal usage hours. The only exception will be if external resources are down.

2.2.4. Ease of Use

    2.2.4.1. The program will be inherently user friendly. Offering drop down menus and search capabilities. It will be visually pleasing and ordered in a way that makes sense. Users will be able to navigate the application with no prior knowledge. using elements from preexisting user interfaces and using an intuitive user flow.

2.2.5. Execution Speed

    2.2.5.1. The application will run at a speed such that it will seem instantaneous

        2.2.5.1.1. Logging in, sending messages, initial searching are the only exceptions

2.2.6. Development environment

    2.2.6.1. The UI will be designed through Android Studio

    2.2.6.2. It will be written in Java or Kotlin, and SQL

    2.2.6.3. Will use a relational database to store data

    2.2.6.4. Will use Azure

2.2.7. Restrictions

    2.2.7.1. There needs to be a way for no user to put up for rent an item that they are currently renting

    2.2.7.2. The system needs to not falsely charge any users credit card

Research and Development:

Development steps:

Continue research – best language, cloud services, best database
Finish UI design – continue working on app UI sketches
Make UML diagrams – need to create a class diagram, ER database design
Begin design within Android Studio – translate sketches into working prototype
Create database – create the database modeled after the ER diagram
Start programming functionality - begin creating Classes and functions

Research:

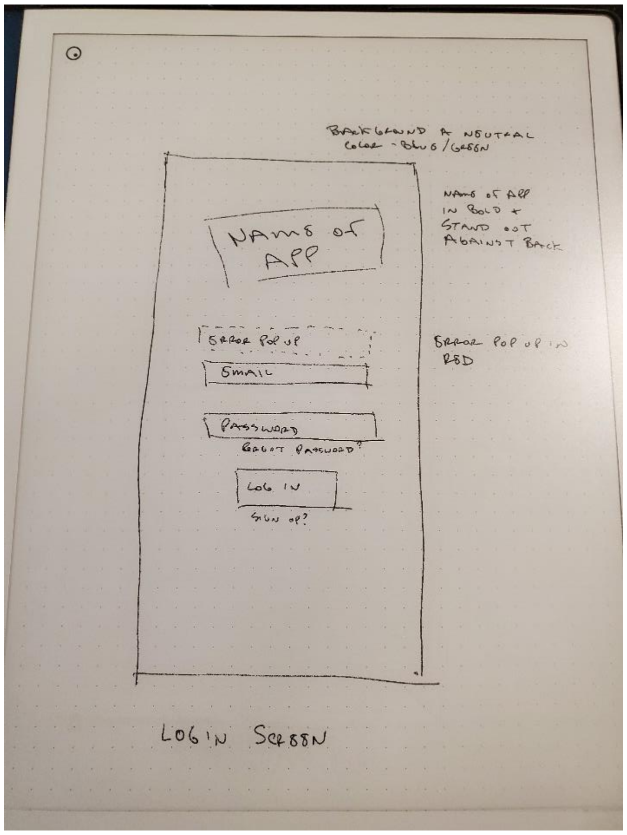
Language: Java vs Kotlin

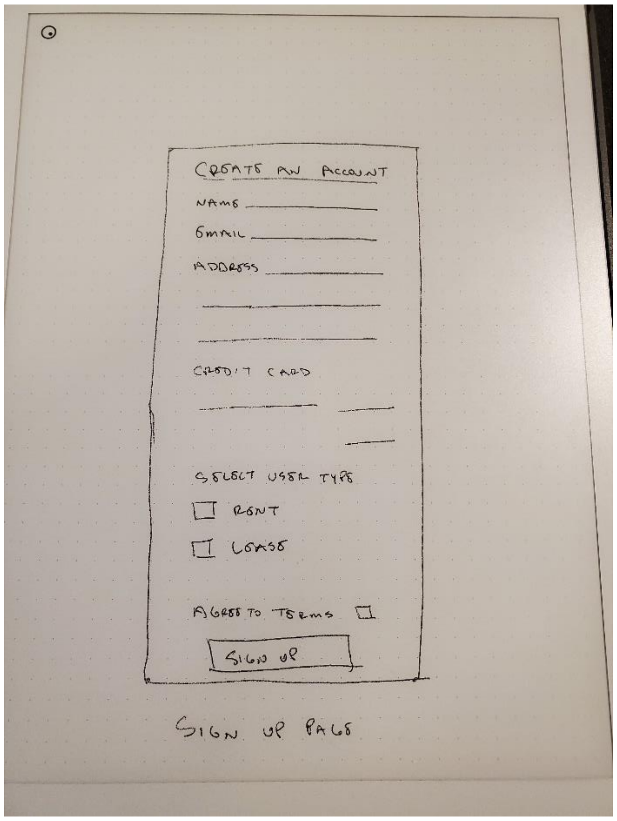
Kotlin is the obvious choice, it is the go to language for Android application development. This project will also give a great opportunity to dive into its wider implementation.

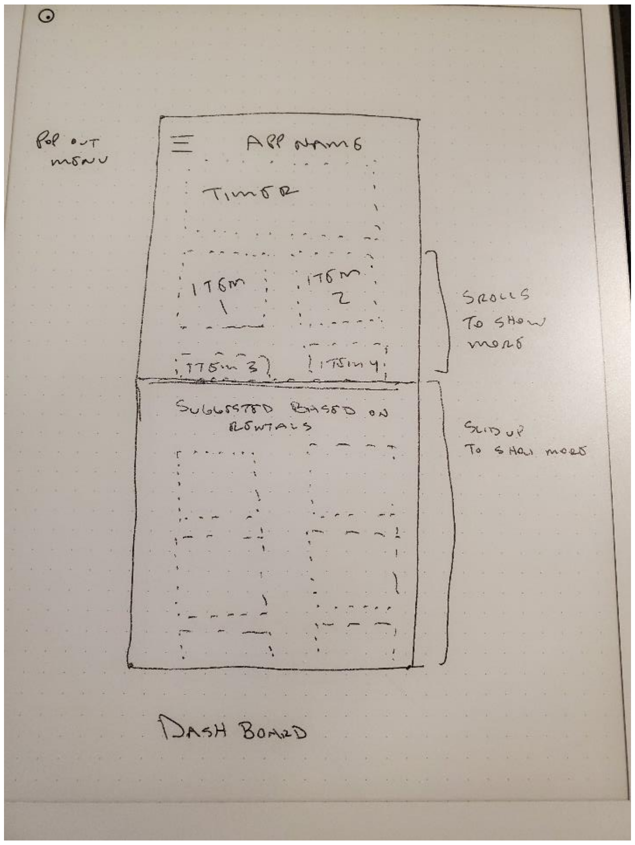
Databases:

SQLite to be used since it is easy to implement and lightweight

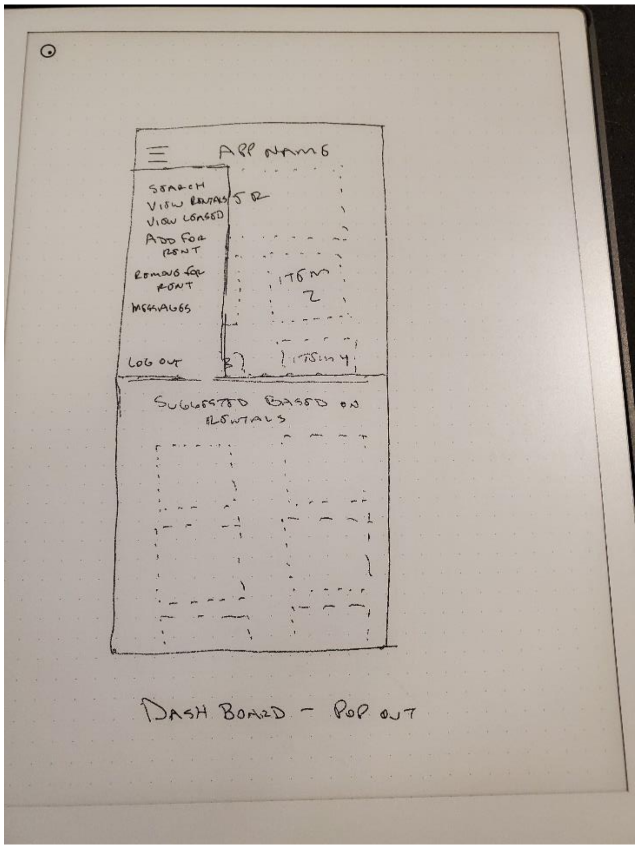
UI Design:


BACKGROUND A NEUTRAL COLOR - BLUE / GREEN
NAME OF APP
NAME OF APP IN BOLD + STAND OUT AGAINST BACK
ERROR POP UP
ERROR POP UP IN RED
EMAIL
PASSWORD
FORGOT PASSWORD?
LOG IN
SIGN UP?
LOGIN SCREEN

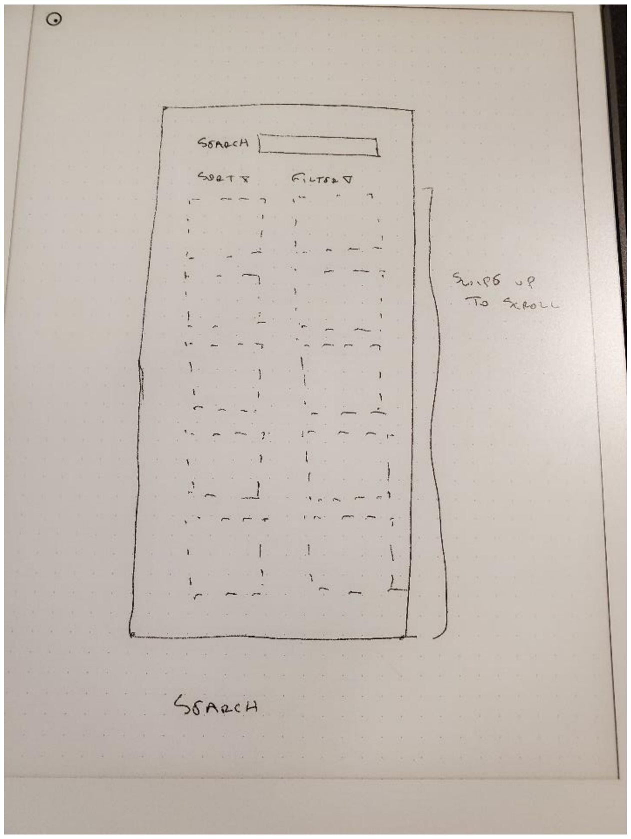

CREATE AN ACCOUNT
NAME
EMAIL
ADDRESS
CREDIT CARD
SELECT USER TYPE
RENT
LEASE
AGREE TO TERMS
SIGN UP
SIGN UP PAGE

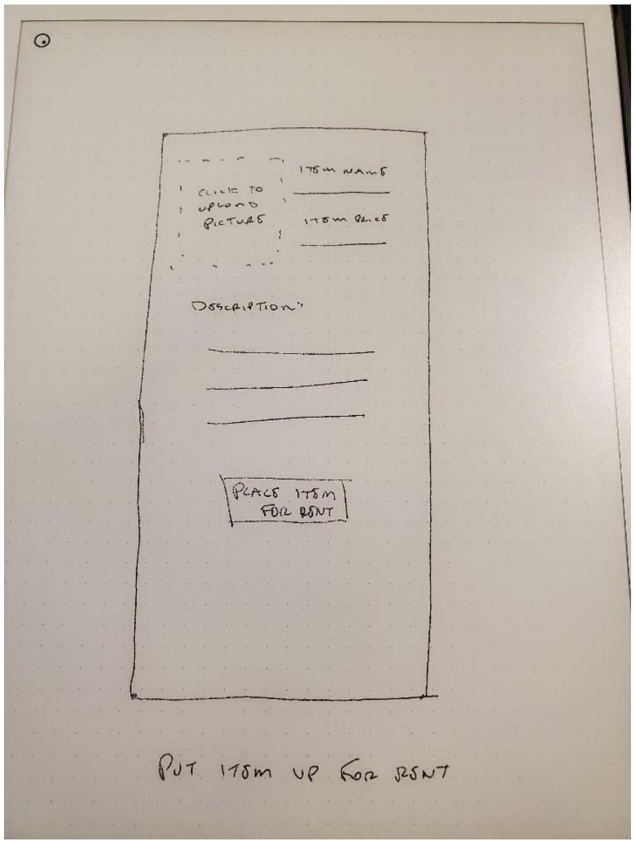


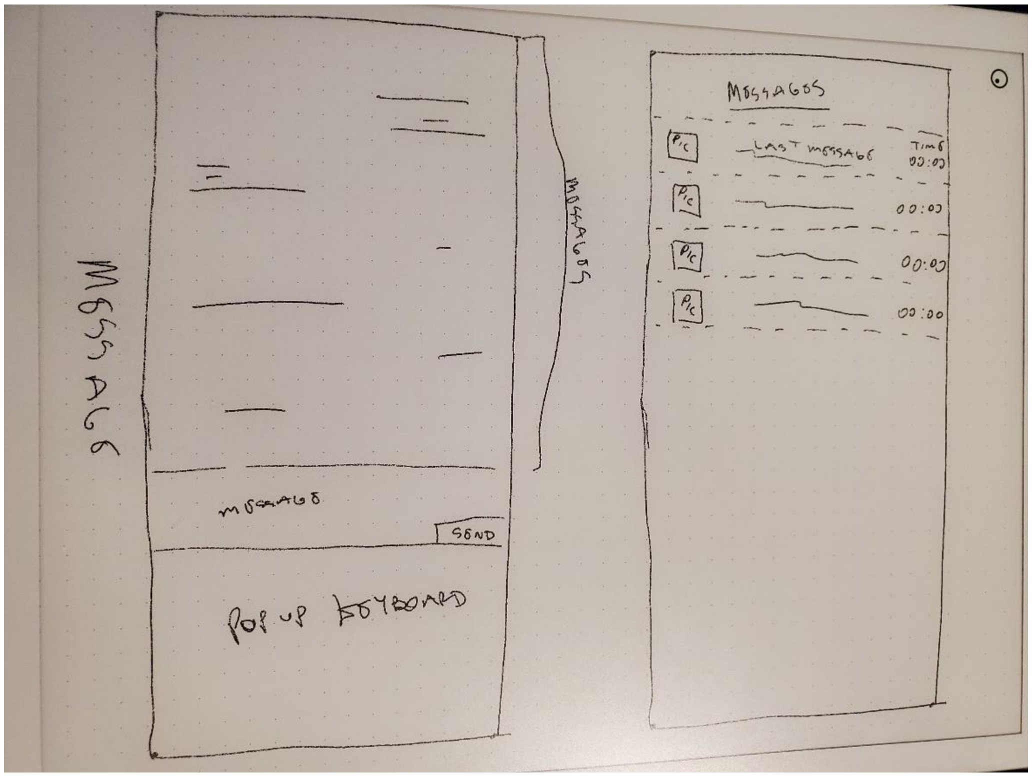
POP OUT MENU
APP NAME
TIMER
ITEM 1
ITEM 2
ITEM 3
ITEM 4
SCROLLS TO SHOW MORE
SUGGESTED BASED ON RENTALS
SLIDE UP TO SHOW MORE
DASH BOARD

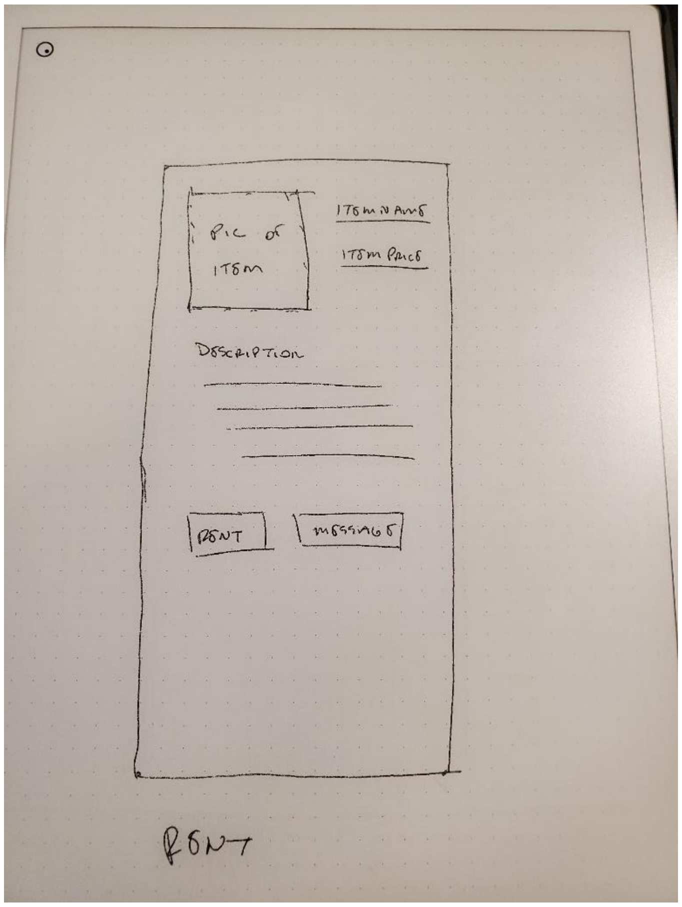


APP NAME
SEARCH
VIEW RENTALS
VIEW LEASED
ADD FOR RENT
REMOVE FOR RENT
MESSAGES
LOG OUT
ITEM 2
ITEM 4
SUGGESTED BASED ON RENTALS
DASH BOARD - POP OUT

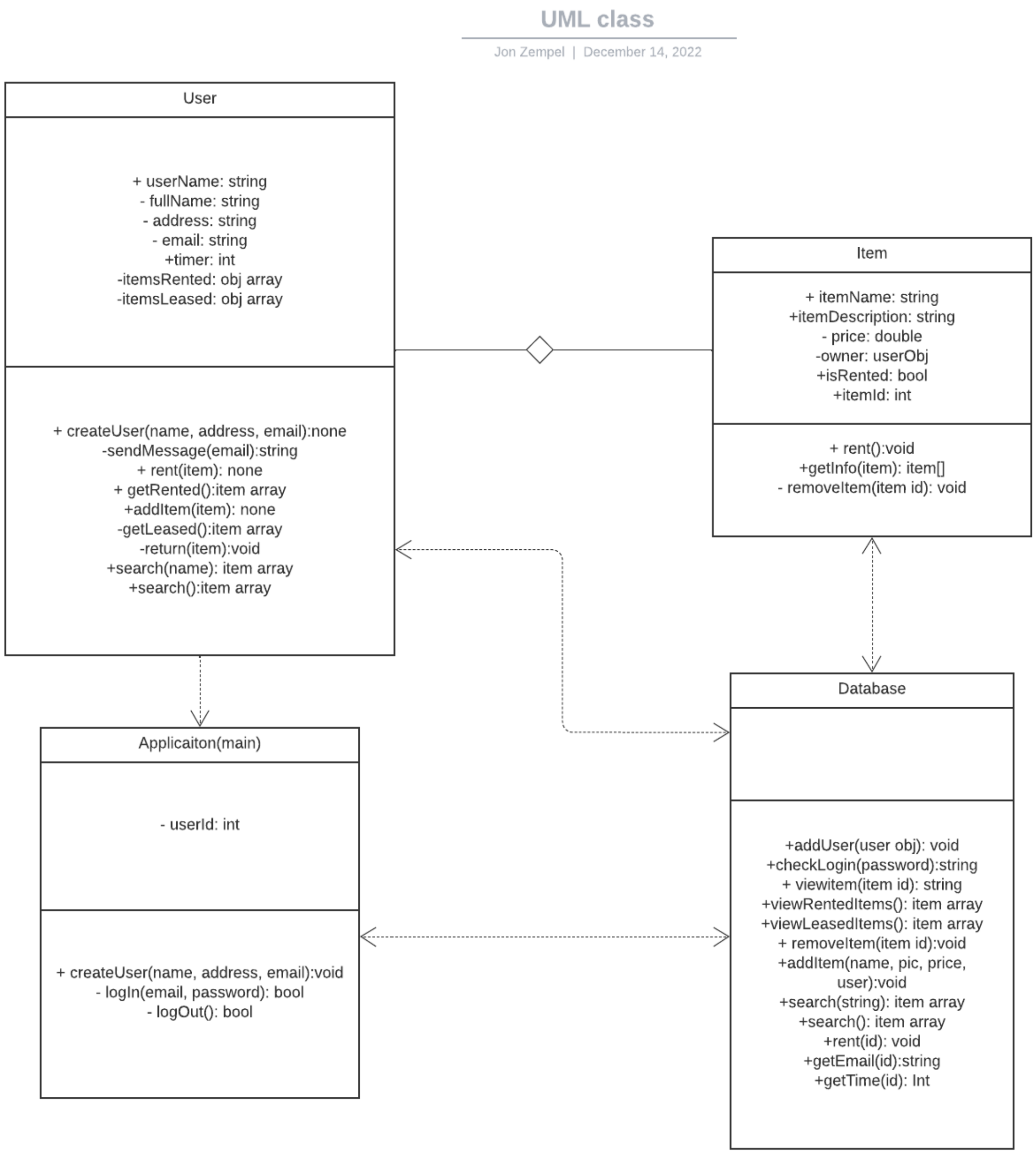

SEARCH
SORT ▽
FILTER ▽
SWIPE UP TO SCROLL
SEARCH

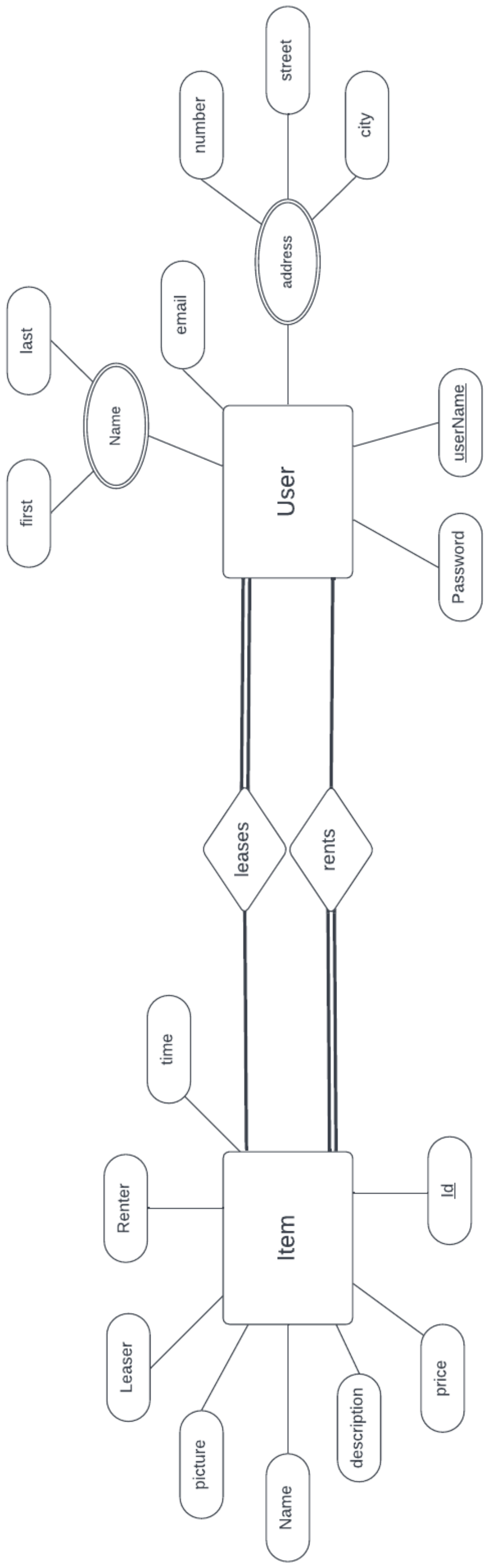

CLICK TO UPLOAD PICTURE
ITEM NAME
ITEM PRICE
DESCRIPTION:
PLACE ITEM FOR RENT
PUT ITEM UP FOR RENT


MESSAGE
MESSAGES
MESSAGE
SEND
POP UP KEYBOARD
MESSAGES
PIC
LAST MESSAGE
TIME
00:00
00:00
00:00
00:00


PIC OF ITEM
ITEM NAME
ITEM PRICE
DESCRIPTION
RENT
MESSAGE
RENT

UML Diagrams:





User
Name
first
last
email
address
number
street
city
userName
Password
leases
rents
Item
time
Renter
Leaser
picture
Name
description
price
Id