CS3354 Software Engineering Final Project Deliverable 2
Group 6
Neighbors.com
Preston Glenn, Sunita Aryal, Connie Bardalez, Ritesh Manandhar,Akshay Durvasula, Ram Panda, Advait

Project Deliverable 1 Content

1. Final Project Draft Description

A. Draft Description:

Our idea

A rental website where users can request to rent items from people who live near them for a short period of time. For example, if someone needs a high-quality camera or speaker system for a day but does not want to own it permanently, they can pay someone in their neighborhood (who has the items) a small fee to borrow the items and then return them.

Our Motivation

We know that in hard economic times, it is very difficult to buy a lot of expensive luxuries at full price. With many people living paycheck-to-paycheck, financing big purchases may not work out or even worse, may end up costing more than the original item in the long run. Therefore, why not borrow your neighbor’s high-quality speaker system for a party, for example, and then give it back a couple of days later for a small fee instead of paying outrageous prices for it.
Even if one is not under economic strain, there are many things that we can enjoy for short periods of time so that we don’t have to have those items sitting in our house collecting dust. Vice versa, if someone happens to have a big-time purchase item such as a flatbed trailer or floodlight that they would like to make money from, Neighbors.com can be their go-to location to rent out their item.

B. Feedback:

Good proposal and fair distribution of tasks to group members. In your final project report (deliverable 2) please make sure to include the following:

- A thorough search to find similar application implementations. Please cite these work using the IEEE citation format provided on Final Project Specifications document.

- Please make sure to differentiate your design from existing similar applications by including extra features into it.

- Please make sure to explicitly specify those differences by comparing your design with those existing similar applications.

C. Response:

We did a brief trial run to see if we could find any applications that were similar to our app within 10 minutes, and came up with a few similar competitors. The main ones we saw were Facebook Marketplace, ebay, and SpareToolz. Though they are somewhat similar applications, we believe that our project is unique enough to stand on its own. However, we will make sure to address the issue of creating a unique project in the second deliverable.

2. GitHub

We created a github repository titled 3354-Neighbors.com on September 23. The github repository can be accessed from https://github.com/preston-Glenn/3354-Neighbors.com. To view the project_scope please take a look at the github repository.

3. Delegation of Tasks

Everyone:
As a group we will convene to work on task 1.5 so that we all have a say in the design of the project. In a similar manner, we will all work on the Architectural design portion of deliverable. And finally, we will all work together to complete the deliverables documents. Depending on the complexity of the unit testing, we may work on it as a group.

Individually:
Everyone will individually create their own github account and write a brief summary of what they did (may be adapted)
Preston:
1.1, 1.2,1.4 Architectural design--*may pull in everyone or individuals to help*, Cost, Effort and Pricing Estimation, and A test plan for your software--may pull in another person once we learn more about it in class.
Ram:
Writing proposal description, Write Functional requirements, Project Scheduling
Sunita:
1.5, Estimated cost of software products
Ritesh:
Writing Non-functional requirements, Estimated cost of hardware products
Connie:
1.6, Cost, Effort and Pricing Estimation, Use case diagram
Advait:
Class diagram, Comparison of your work with similar designs.
Akshay:
Sequence diagrams, Estimated cost of personnel

4. Software Process Model

The software process model that is employed in the project is agile methodology simply because this model ensures a proper channel of communication, which helps both the clients and the app developers to execute the desired application. This model lets us return to the previous phase to fix any errors, it is cost effective, and running software is done in a shorter time period. Furthermore, customers get to respond in each phase. Another advantageous factor of using this method is that our rental application will have daily maintenance to minimize any bugs when customers are using it. Also, it will be easier to make any updates in order to add new features according to the customer's needs.

5. Software Requirements

A. Functional Requirements

· Users should be able to login to the system and see and set up their full profile information and change their information at any time.

· User should be able to browse and filter through listings for items they desire to rent

· Transactions should be approved by both renter and lender at the time of pickup of items.

· Lenders should have the ability to add a negotiable price and photos of the item when listing it.

· Application should facilitate adequate communication between renter and lender.

B. Non-Functional Requirements

· Product Requirements

- o Efficiency
  - § App should not take more than 4 seconds to start up
  - § Users should not have to wait more than 3 seconds to load items
  - § Application size should not exceed 200 MB
  - § Performance of the application shouldn't change drastically if more users are using it at the same time
- o Usability
  - § Layout of application should be user friendly
  - § Users should be able to locate features and navigate throughout the app with ease
  - § Provide recommendations for users based on previous purchases or searches

- Dependability
    - § Users should receive confirmation via email or text after each purchase
    - § System shouldn’t experience many crashes especially during the payment process
- Security
    - § Personal and payment information of every user shouldn’t be accessible to unauthorized personnel
    - § Users should not be able to modify any important or critical data on the app.

· Organizational Requirements

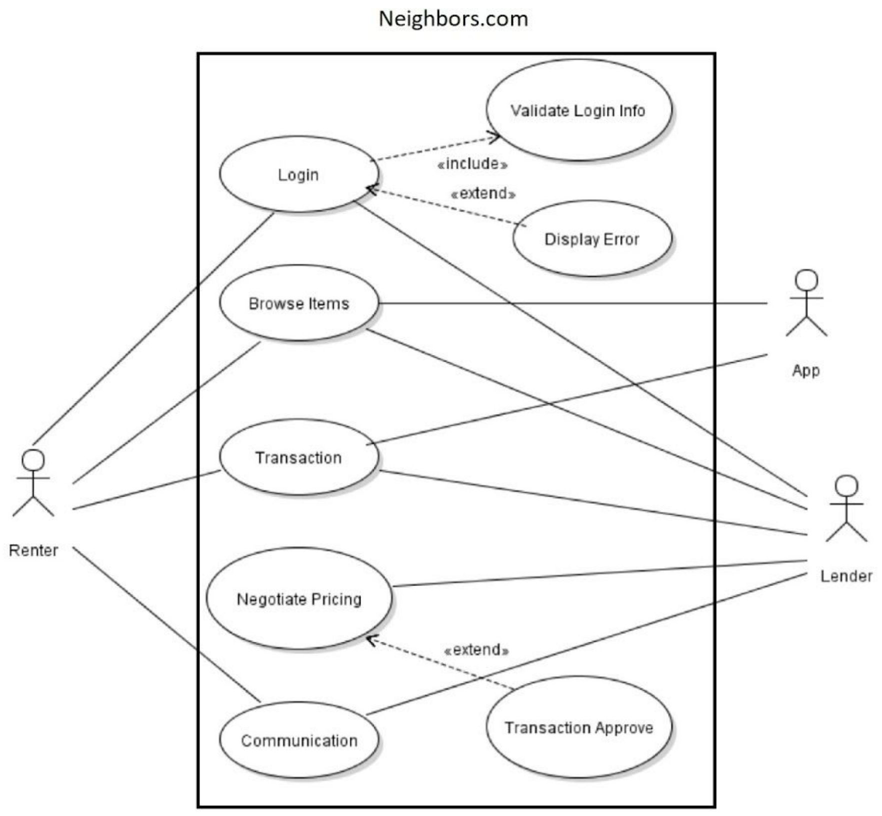
- Environmental
    - § App must be able to adapt to different environments and configurations
- Operational
    - § Must be compatible with iOS and android devices.
    - § Available on App store and play store
    - § IOS 10 or above and android 4.1 or above.
    - § App must be scalable to accommodate any large growth of users
- Development
    - § Any future modifications or updates should be simple and cost efficient

· External Requirements

- Regulatory
    - § Prohibited items such as drugs, alcohol, weapons, or adult content should not be listed
- Ethical
    - § Users should not list inaccurate prices or prices which they don’t intend to sell
    - § Users shouldn’t harass or threaten other users through messages

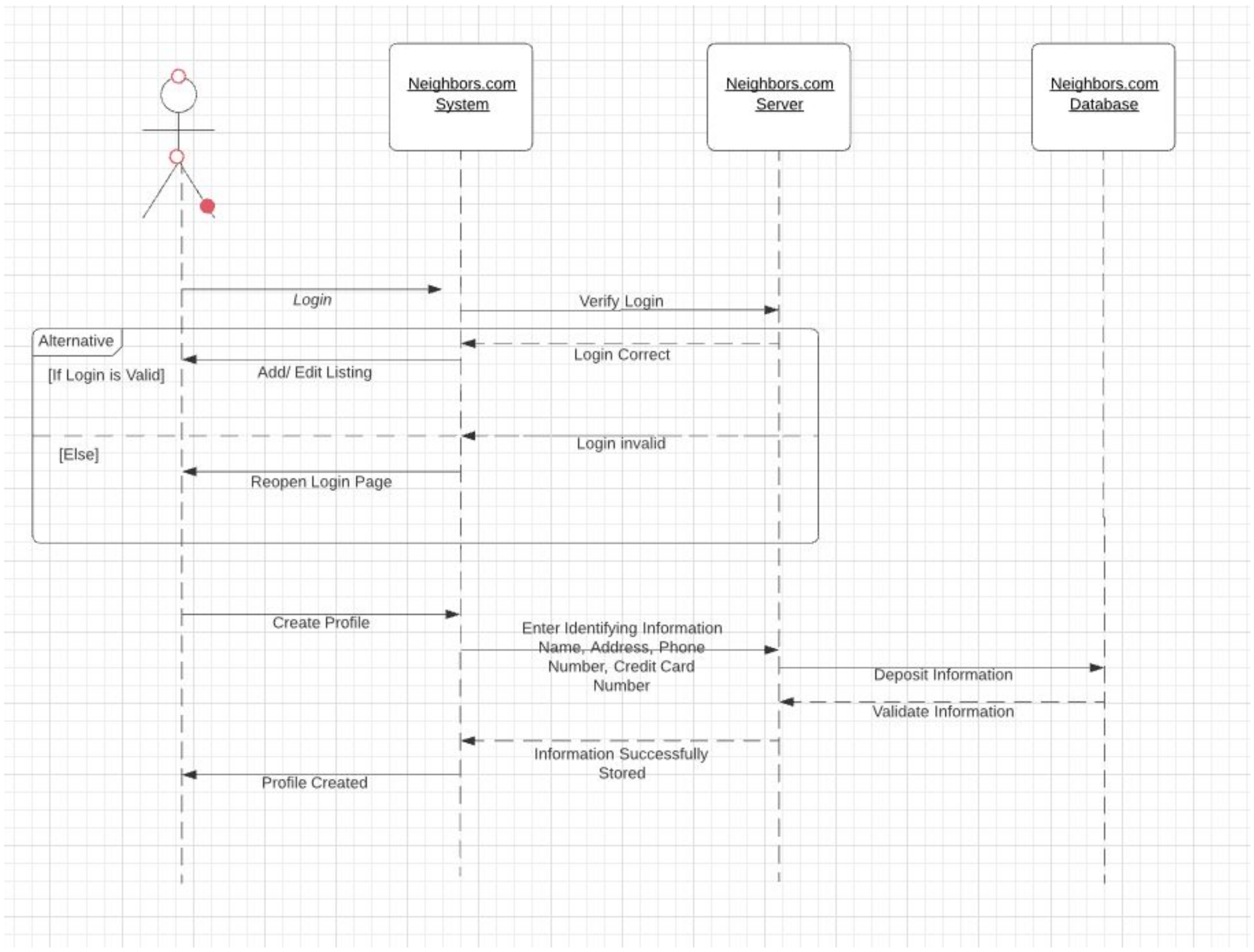
- Legislative
    - § Stolen or illegal items cannot be listed in the application
    - § Users should not post messages that violate rights or privacy of other users
    - § Harvesting user info is prohibited

6. Use Case Diagram

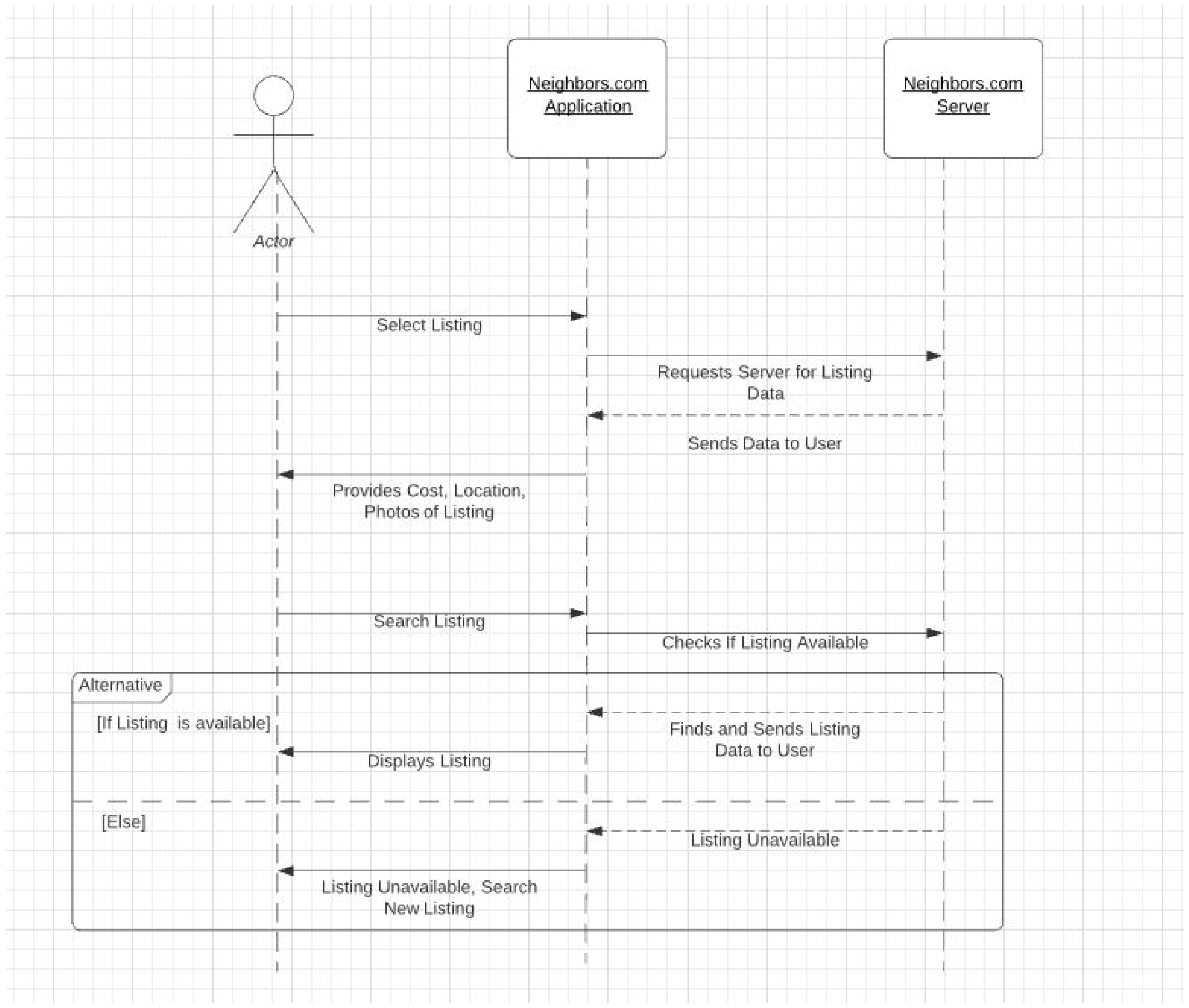


7. Sequence Diagram

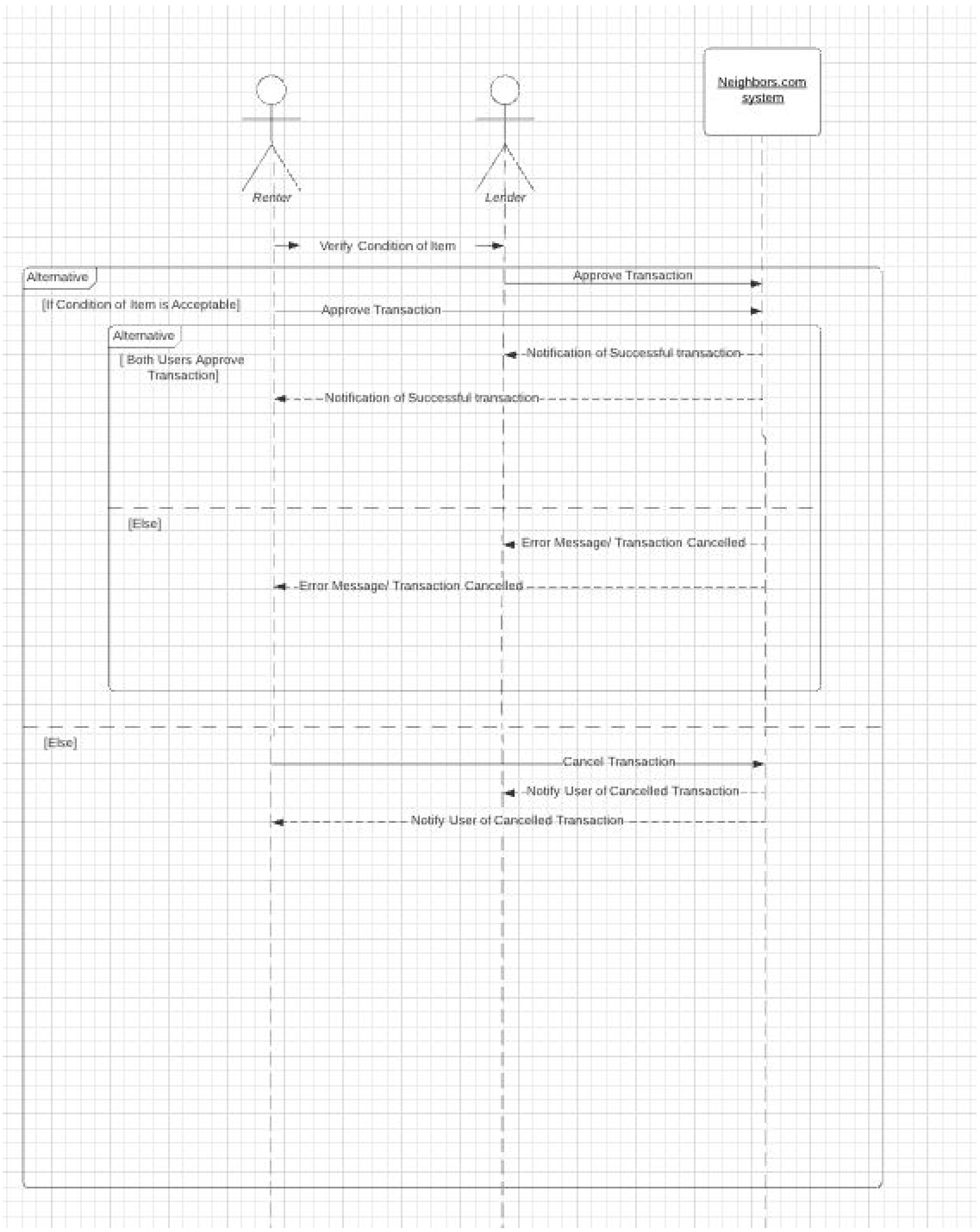
Login-



Browse Listing-

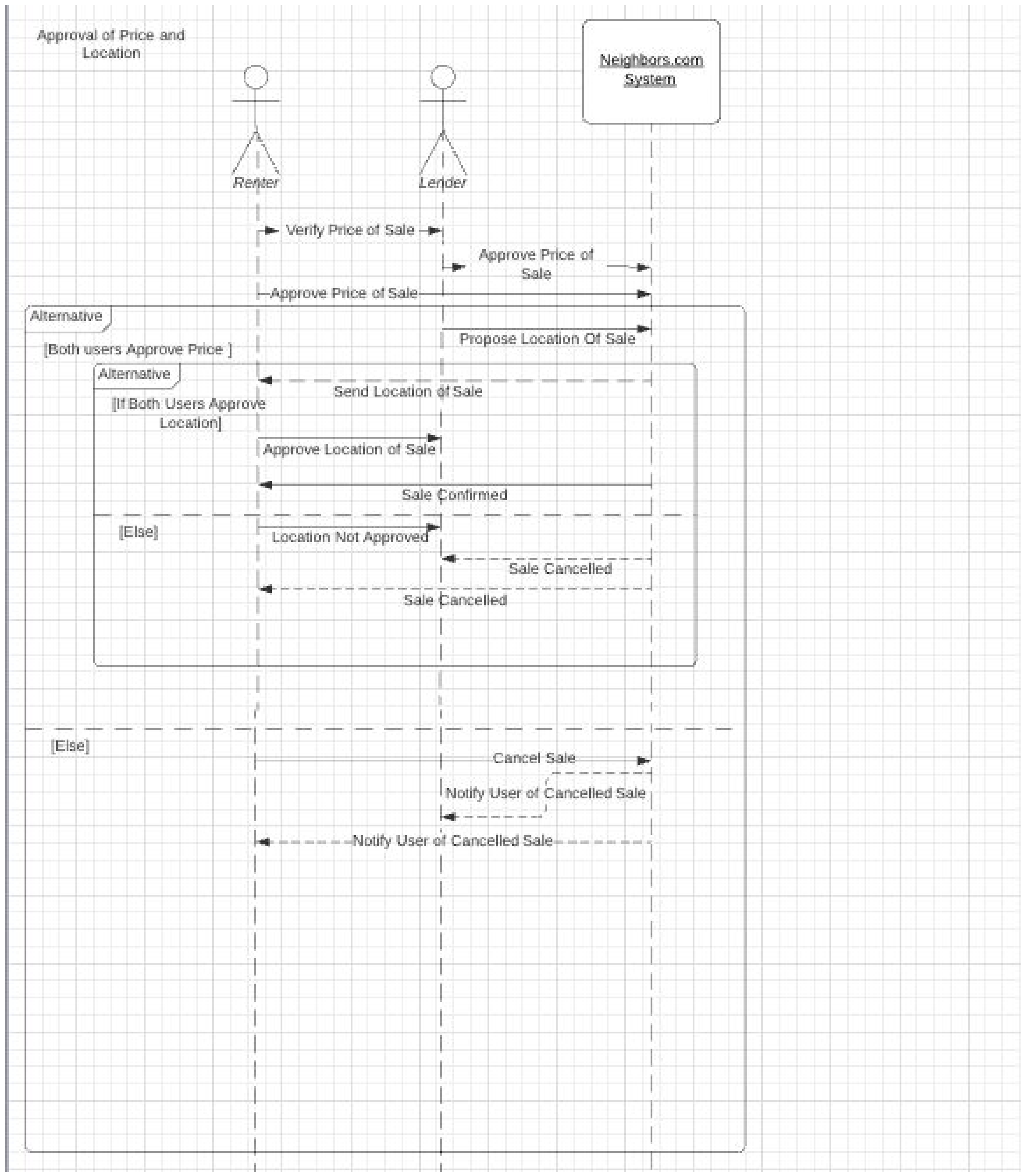

Actor
Neighbors.com Application
Neighbors.com Server
Select Listing
Requests Server for Listing Data
Sends Data to User
Provides Cost, Location, Photos of Listing
Search Listing
Checks If Listing Available
Alternative
[If Listing is available]
Finds and Sends Listing Data to User
Displays Listing
[Else]
Listing Unavailable
Listing Unavailable, Search New Listing

Transaction-

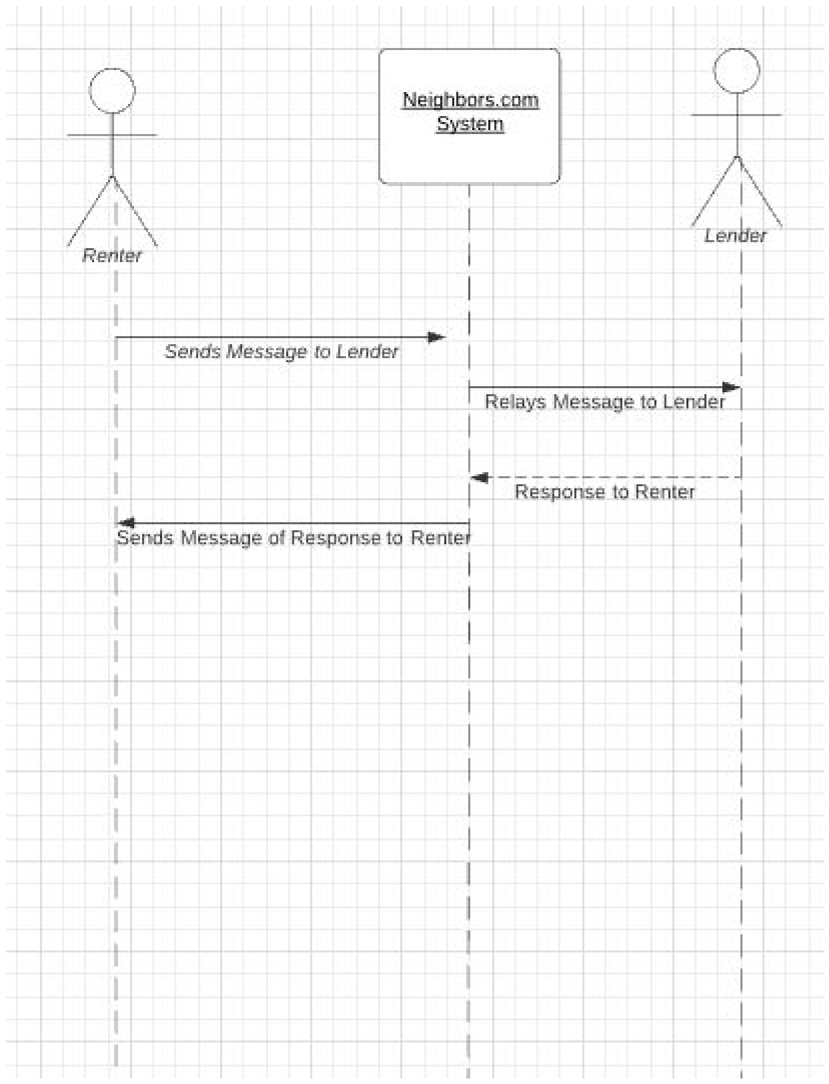


Neighbors.com system
Renter
Lender
Verify Condition of Item
Alternative
Approve Transaction
[If Condition of Item is Acceptable]
Approve Transaction
Alternative
[Both Users Approve Transaction]
Notification of Successful transaction
Notification of Successful transaction
[Else]
Error Message/ Transaction Cancelled
Error Message/ Transaction Cancelled
[Else]
Cancel Transaction
Notify User of Cancelled Transaction
Notify User of Cancelled Transaction

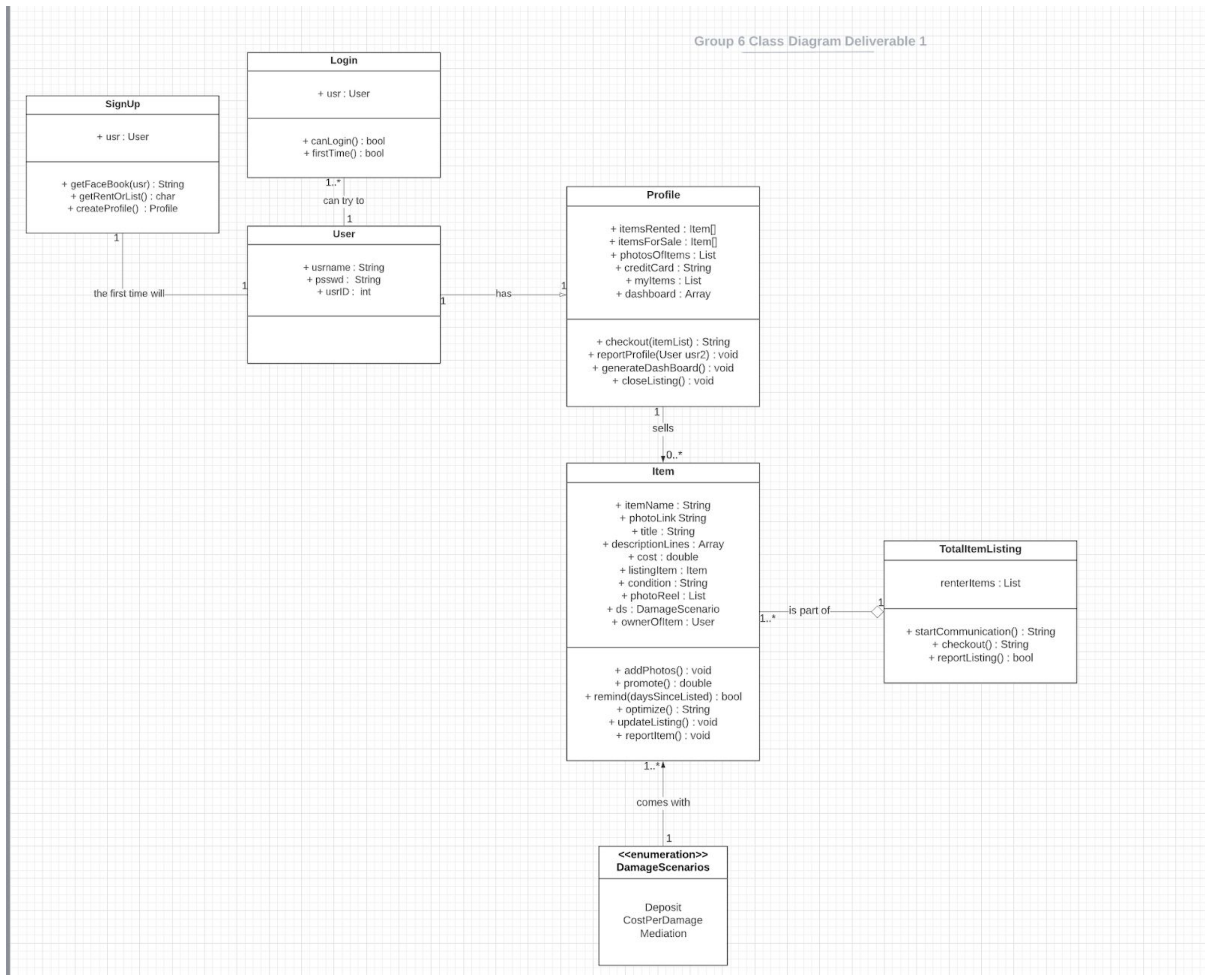
Approval of Price, Location-



Communication



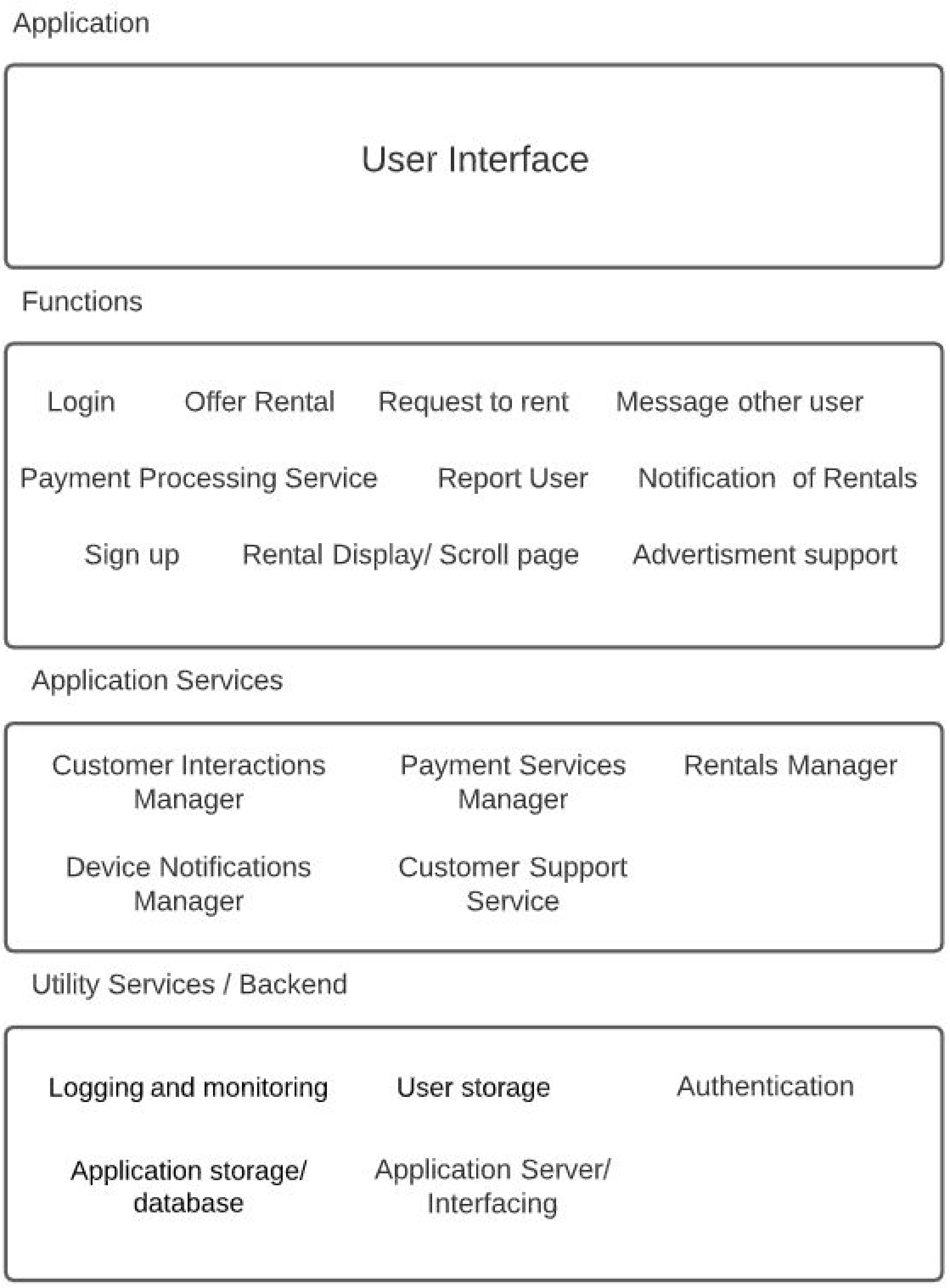
8. Class diagram



LINK PROVIDED IF IMAGE IS DIFFICULT TO SEE

https://lucid.app/documents/view/139e98d7-63a2-4ced-bcea-e650d374480c

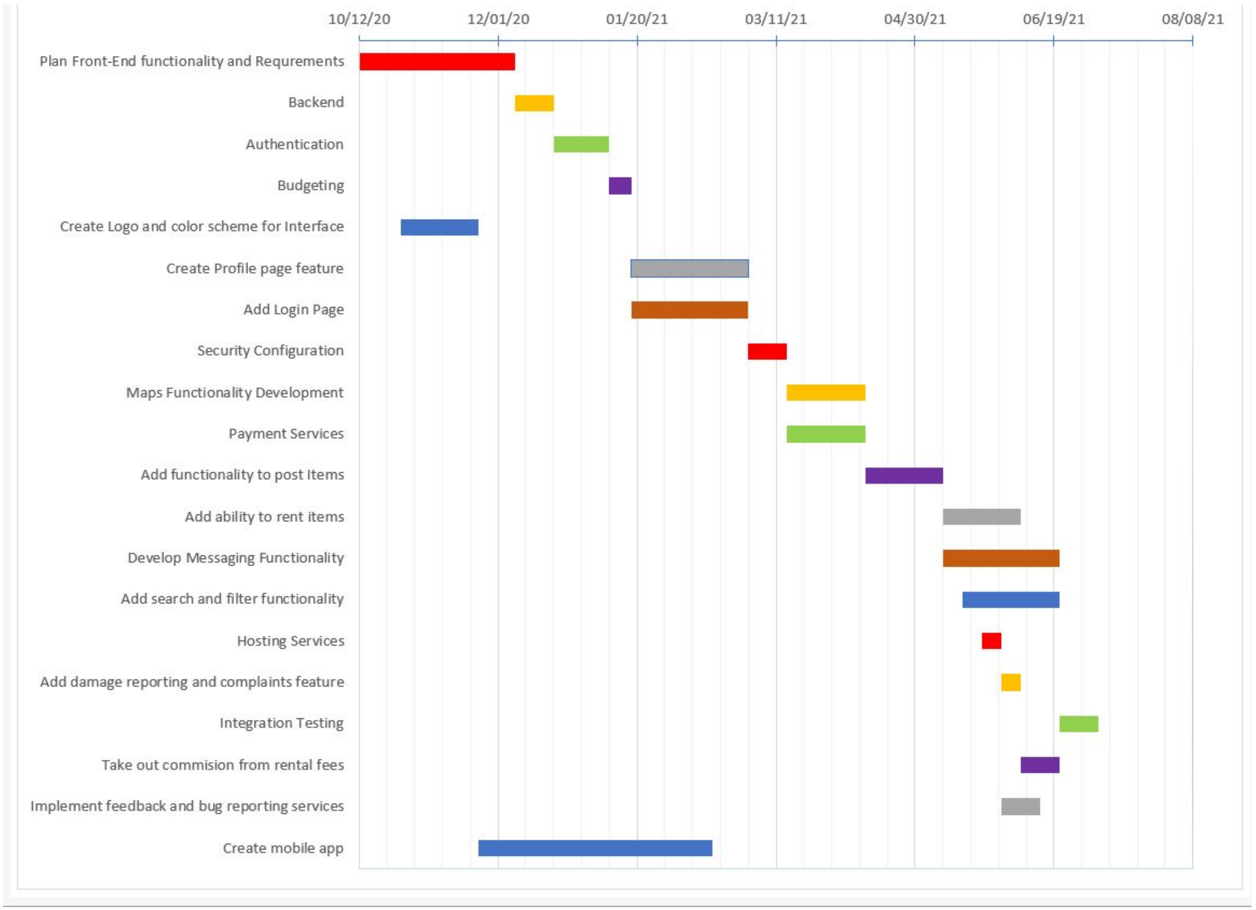
9. Layered Architectural Design

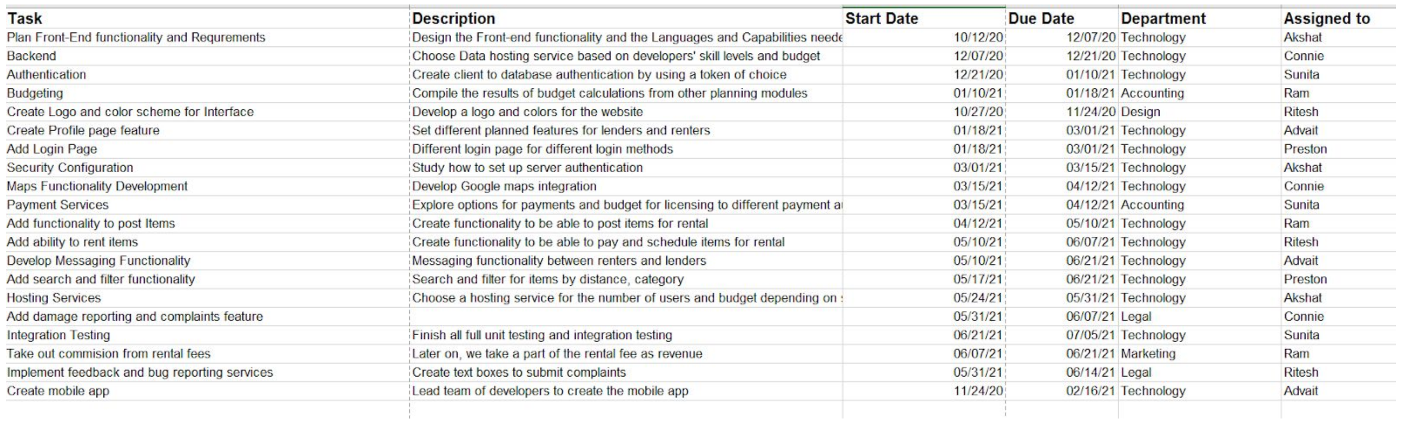


For this software proposal, we decided to go with a layered architectural design approach. We found that its top-down approach meshed well with the fact that our app would have significant user interaction due to the fact that we could have a layer/ “component” dedicated to the user interface. We also liked the fact that each layer acted as an abstract implementation, where each one could be worked on and refactored and you would have no impact on the rest of the app. This would be highly important for a startup esq company which would be striving for rapid implementation of improvements wherever they can be made. Not only can each layer be replaced, but each as long as the interface for each layer is the same, each layer can be worked on independently of the other. This enables more focused teams, where two separate teams can be created, where one focuses on the frontend and one works on the backend.

Project Deliverable 2 Content

1. Project Scheduling Estimation Chart





Task	Description	Start Date	Due Date	Department	Assigned to
Plan Front-End functionality and Requrements	Design the Front-end functionality and the Languages and Capabilities neede	10/12/20	12/07/20	Technology	Akshat
Backend	Choose Data hosting service based on developers' skill levels and budget	12/07/20	12/21/20	Technology	Connie
Authentication	Create client to database authentication by using a token of choice	12/21/20	01/10/21	Technology	Sunita
Budgeting	Compile the results of budget calculations from other planning modules	01/10/21	01/18/21	Accounting	Ram
Create Logo and color scheme for Interface	Develop a logo and colors for the website	10/27/20	11/24/20	Design	Ritesh
Create Profile page feature	Set different planned features for lenders and renters	01/18/21	03/01/21	Technology	Advait
Add Login Page	Different login page for different login methods	01/18/21	03/01/21	Technology	Preston
Security Configuration	Study how to set up server authentication	03/01/21	03/15/21	Technology	Akshat
Maps Functionality Development	Develop Google maps integration	03/15/21	04/12/21	Technology	Connie
Payment Services	Explore options for payments and budget for licensing to different payment a	03/15/21	04/12/21	Accounting	Sunita
Add functionality to post Items	Create functionality to be able to post items for rental	04/12/21	05/10/21	Technology	Ram
Add ability to rent items	Create functionality to be able to pay and schedule items for rental	05/10/21	06/07/21	Technology	Ritesh
Develop Messaging Functionality	Messaging functionality between renters and lenders	05/10/21	06/21/21	Technology	Advait
Add search and filter functionality	Search and filter for items by distance, category	05/17/21	06/21/21	Technology	Preston
Hosting Services	Choose a hosting service for the number of users and budget depending on	05/24/21	05/31/21	Technology	Akshat
Add damage reporting and complaints feature		05/31/21	06/07/21	Legal	Connie
Integration Testing	Finish all full unit testing and integration testing	06/21/21	07/05/21	Technology	Sunita
Take out commision from rental fees	Later on, we take a part of the rental fee as revenue	06/07/21	06/21/21	Marketing	Ram
Implement feedback and bug reporting services	Create text boxes to submit complaints	05/31/21	06/14/21	Legal	Ritesh
Create mobile app	Lead team of developers to create the mobile app	11/24/20	02/16/21	Technology	Advait

Total Project time: 32 weeks

Justification for project Scheduling:

I have made a liberal estimate that creating a solid, working Minimum Viable product will take 36 weeks. I estimated this time frame based on the assumption that our team consists of 15 developers working full time. I measured all tasks as a measure of 2-week-long sprint cycles. I assumed we will not be working on the weekends and as such, each sprint cycle will consist of 10 days of work.
As an example of why I estimated the project to take a total of 9 months (36 weeks). One of the tasks is to integrate google maps functionality into the web application. I know from previous project experience that this in itself will take a lot of time. This is because a team of 5 people took over 4 weeks just to implement the basic functionality of Google Maps in an app. Therefore, it will take us over 3 sprint cycles to program in the google maps functionality, add in features such as plotting renter locations, searching for closeby renters, etc.

2. Function Point Complexity Analysis

Function Point

Func. Category	Count		Simple	Average	Complex		Count X Complexity
# of User Input	5		3	4	6		15
# of user output	3		4	5	7		12
# of user queries	4		3	4	6		16
# of data tables	4		7	10	15		60
# of external interfaces	5		5	7	10		50

GFP = 15 + 12 + 16 + 60 + 50 = 143
Productivity = 10

Determine processing complexity (PC).
(1) Does the system require reliable backup and recovery? 3
(2) Are data communications required? 4
(3) Are there distributed processing functions? 0
(4) Is performance critical? 1
(5) Will the system run in an existing, heavily utilized operational environment? 5
(6) Does the system require online data entry? 5
(7) Does the online data entry require the input transaction to be built over multiple screens or operations? 4
8) Are the master files updated online? 3
(9) Are the inputs, outputs, files, or inquiries complex? 5
(10) Is the internal processing complex? 3
(11) Is the code designed to be reusable? 2
(12) Are conversion and installation included in the design? 3
(13) Is the system designed for multiple installations in different organizations? 2
(14) Is the application designed to facilitate change and ease of use by the user? 5

Compute processing complexity adjustment (PCA).
PCA = 0.65 + 0.01 x (1 + 2 x 2 + 3 x 3 + 2 x 4 + 4 x 5) = 1.07

Compute function point (FP) using the formula: FP = GFP × PCA
FP = 143 x 1.07 = 153.01

Effort = FP/productivity = 153.01/ 10 = 15 person-weeks

Estimated Price = hardware products + software products + cost personnel
Estimated Price = 600,555.15

3. Estimated cost of hardware products (such as servers, etc.)

	Description	**Price**	**Quantity**	**Total**
Laptops	i5 14in 8GB mem 256GB SSD	$899 [4]	15	$14,597.51
Monitors	24-inch FHD	$109 [5]	5	$589.96
Mouse	2.4 wireless mouse	$8.66 [6]	15	$140.62
Router	Dual Band Gigabit Wireless router	$59.99 [3]	1	$64.94
				$15,393.03

For other hardware products such as servers and storage we decided that instead of spending thousands of dollars purchasing them, cloud server and storage would be a good alternative. We included the pricing of the cloud servers in the software cost portions.

4. Estimated cost of software products (such as licensed software, etc.)

The storage and server prices are based on the assumption that we would have 50,000 users in the first couple of months. These prices also take into account that not all users will be posting pictures and utilizing many functions of the app on a daily basis.

Software Product Cost:

Authentication Service 50,000 users	Phone authentication - 400/month Since our project will be 32 weeks long therefore 8 months *400 = $3,200 total
Intellij IDE Subscription	$499 *15 per user, yearly = $7,485
Realtime Database	$10 per month [2] $10 x 8 months = $80
Storage	$33.58 per month [2] $33.58 x 8 months = $268.64
Cloud Functions	$41.06 per month [2] $41.06 x 8 months = $328.48
iOS Development	a simple iOS app like ours with basic functionality thats takes up almost about two months to build will cost about $30,000
Total	$41,362.12

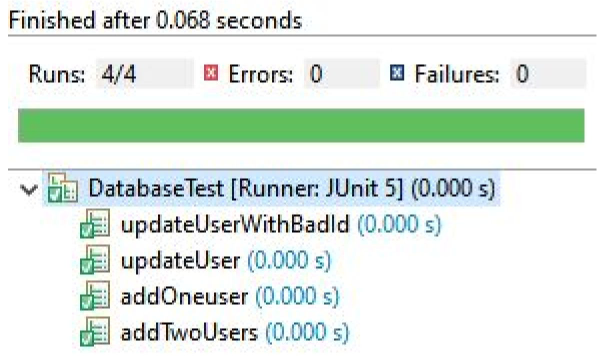
5. Estimated cost of personnel

Software product cost estimate:	**Roles**	**Subscriptions**	**Cost**
Marketing and Product management	Provides app with updates and customer support.	$35 hourly 1 person	32 weeks project 32 weeks * 40 hours/week = 1280 hours 1280 * 35- * **$44800**
Unit and integration testing	Engineers monitor every phase of the software development process. QA managers make sure that new products are bug-free.	$25 hourly 4 people	4 weeks project 4 weeks * 40 hours/week = 160 hours 160 hours * $25 * 4 = **$16000**
UI/UX designer	person on which depends on the app structure and visual appearance.	$25 hourly 2 people	32 weeks project 32 weeks * 40 hours/week = 1280 hours 1280 * $25 * 2 = **$64000**
Backend and database maintenance including storage	creates the app back-end infrastructure and API integrations.	$ 30 hourly 2 people	10 weeks project 10 weeks * 40 hours/week = 400 hours 400 hours * $30 * 2 = **$24000**
Developers	Range from $50 to $250 per hour	$50 hourly 6 people	32 weeks project 32 weeks * 40 hours/week = 1280 hours 1280 * $50 * 6 = $**384000**
Miscellaneous Cost			**$2,000**
Total		15 people total	**$543800**

6. Software Testing Plan

Since we went with a layered architecture approach so that we could refactor our code easily, we believe that utilizing a bottom-up integrated testing approach would be conducive to our app development. The reason is that anytime we refactor a module we can simply worry about the one of the bottom atomic modules and not have to worry about updating any other tests outside of the changed component.

As a demonstration for the test plan of this project, we wrote a couple classes that represent a user and a database of users, or in this case an arraylist of users. In the code, there is the capability to update the user, add a user, and to print the users in the DB. Also, the user class has the basic get and set methods for name and id. For this demo, we will simply be testing the addUser method and the updateUserName() function. For the add method, since the input is limited to a user object, the only test case we have to do is adding 1 user and then adding 2 users to the “database” and then verifying that it added it correctly. For the updateUserName() function, we need to validate that it works as it is supposed to and then also, that an empty string won’t update the Database.


```
Finished after 0.068 seconds

Runs: 4/4     Errors: 0     Failures: 0

DatabaseTest [Runner: JUnit 5] (0.000 s)
    updateUserWithBadId (0.000 s)
    updateUser (0.000 s)
    addOneuser (0.000 s)
    addTwoUsers (0.000 s)
```

7. Comparison of your work with similar designs.

The closest real-world software comparison to our idea would be the highly popular Facebook Marketplace and eBay. Both platforms allow users to buy and sell virtually any item they own. Since the marketplace's social media counterpart, Facebook, already creates a full profile for users, the marketplace uses the same profiles for its website. Neighbors.com introduces a key change to the framework of the facebook marketplace and eBay. Namely, customers do not *buy* an item, they *rent* it. This creates another layer of sophistication that our system has to manage for returning items back to users. Also, Neighbors.com only features items from lenders that are exclusively within a certain radius of the user's location while facebook marketplace and eBay also show items from out of the set radius.(as in, items that are sold outside a specified radius can still be purchased).

8. Conclusion

When we first started this project, we didn’t really know what we wanted to do. We had several ideas, but none of them seemed to be the perfect fit for this project. Eventually, we came to the conclusion to create a rental application among neighbors. We decided this for several reasons. First of all, though there are apps that serve familiar purposes, there didn’t seem to exist a large platform that attempts to service renting items amongst neighbors. Some small applications that were similar only had specific items like tools while we wanted an app that offers a wide variety of products. And we also thought that this project could be designed and planned within the timeframe given, so we found it was a win win.

As we started to progress with the project though, we found at first that though we had the central idea to the app, we didn’t really know what the user experience would be and how it would immediately function. This issue was resolved when we completed task 1.5 and wrote out the project scope. This really had us sit down and better understand everything that we would need to implement if we were to create this app in reality. As a result, we spent hours exploring the best features/methods to implement for the app and compared them to features used by similar applications. For example, when we knew that we needed users to be able to communicate with each other in order to exchange the item, however, at first we forgot to consider the fact of how customers would actually approve the transaction and indicate that they had received/given out the item that was being rented. In the end, we faced a few realization of potential dangers of some of our ideas which in turn led to several small changes that helped the concept of our application become more stable.

Over the course of this project, we found that many questions would arise of how we wanted a specific issue to be resolved. And so over time we found that though changes would be made to the design of the project, the scope of the project would not change. It was just that the scope at which we were looking at the problem/project would change and thus require further attention.

References

[1] I. J. “Buy IntelliJ IDEA Ultimate: Pricing and Licensing, Discounts - JetBrains Toolbox Subscription,” JetBrains, 2020. [Online]. Available: https://www.jetbrains.com/idea/buy/. [Accessed: 06-Nov-2020].

[2] F. “Firebase Pricing,” Google. [Online]. Available: https://firebase.google.com/pricing. [Accessed: 07-Nov-2020].

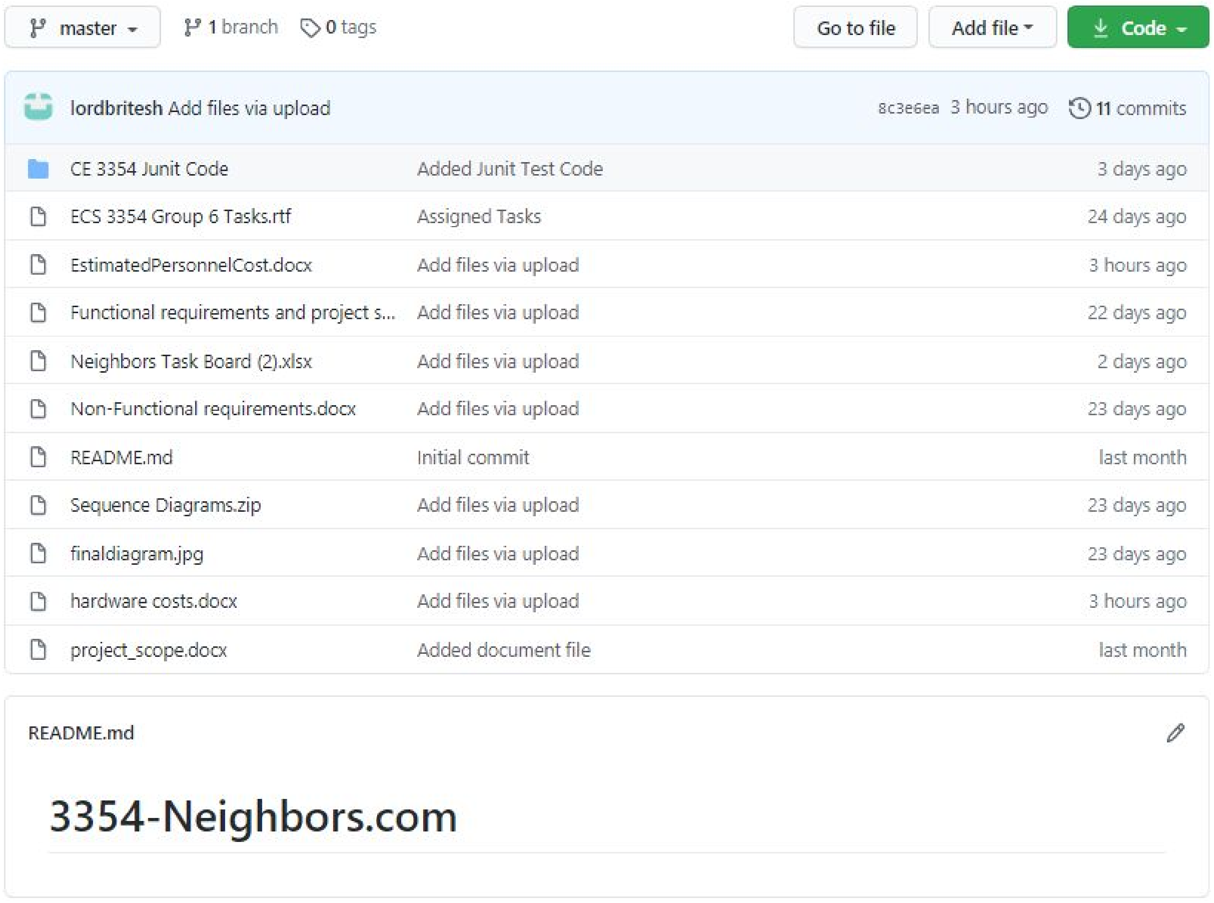
[3] “TP-Link AC1750 Smart WiFi Router (Archer A7) - Dual Band Gigabit Wireless Internet Router for Home, Works with Alexa, VPN Server, Parental Control and QoS,” Amazon.com. [Online]. Available: https://www.amazon.com/TP-Link-AC1750-Smart-WiFi-Router/dp/B079JD7F7G/ref=sr_1_4?dchild=1&amp;keywords=router&amp;qid=1604864029&amp;sr=8-4. [Accessed: 07-Nov-2020].

[4] “Lenovo ThinkPad T490 Laptop - 14’ FHD - 1.6 Ghz Intel Core i5-8365U Quad-Core - 8GB - 256GB SSD - Windows 10 pro - 3YR Premier,” *Amazon.com*. [Online]. Available: https://www.amazon.com/Lenovo-ThinkPad-T490-i5-8365U-Quad-Core/dp/B081B92Y9R/ref=sr_1_15?dchild=1&keywords=i5+lenovo&qid=1604864530&refinements=p_36%3A89000-90000&rnid=2421885011&sr=8-15. [Accessed: 07-Nov-2020].

[5] “HP 24mh FHD Monitor - Computer Monitor with 23.8-inch IPS Display (1080p) - Built-in Speakers and VESA Mounting - Height/Tilt Adjustment for Ergonomic Viewing - HDMI and DisplayPort,” *Amazon.com*. [Online]. Available: https://www.amazon.com/HP-24mh-FHD-Monitor-Built/dp/B08BF4CZSV/ref=sxin_10_ac_d_rm?ac_md=0-0-bW9uaXRvcg%3D%3D-ac_d_rm&cv_ct_cx=monitor&dchild=1&keywords=monitor&pd_rd_i=B08BF4CZSV&pd_rd_r=f43f0e6c-246a-46ba-ba1b-0ece724d86ec&pd_rd_w=zVth1&pd_rd_wg=rbm3M&pf_rd_p=3d1a8341-be16-45b1-ae3d-ba8c533ec9f0&pf_rd_r=N4Y0A7W208X5AAJXY5XB&psc=1&qid=1604864822&sr=1-1-12d4272d-8adb-4121-8624-135149aa9081. [Accessed: 07-Nov-2020].

[6] “Jelly Comb 2.4G Slim Wireless Mouse with Nano Receiver MS001,” *Amazon.com*. [Online]. Available: https://www.amazon.com/Jelly-Comb-Wireless-Mouse-Receiver/dp/B076F5P28T/ref=sr_1_4?dchild=1&keywords=mouse&qid=1604864982&s=electronics&sr=1-4. [Accessed: 07-Nov-2020].

9. GitHub requirement:



lordbritesh Add files via upload 8c3e6ea 3 hours ago 11 commits

CE 3354 Junit Code	Added Junit Test Code	3 days ago
ECS 3354 Group 6 Tasks.rtf	Assigned Tasks	24 days ago
EstimatedPersonnelCost.docx	Add files via upload	3 hours ago
Functional requirements and project s...	Add files via upload	22 days ago
Neighbors Task Board (2).xlsx	Add files via upload	2 days ago
Non-Functional requirements.docx	Add files via upload	23 days ago
README.md	Initial commit	last month
Sequence Diagrams.zip	Add files via upload	23 days ago
finaldiagram.jpg	Add files via upload	23 days ago
hardware costs.docx	Add files via upload	3 hours ago
project_scope.docx	Added document file	last month

README.md

3354-Neighbors.com