SW Engineering CSC 648/848 Fall 2022

Project: Bill Splitting Web App

FunBill

Team 01 = Team Fun

Denyse Luzon	Team Lead, dluzon1@sfsu.edu
Bhagdeep Sandhu	Database Master
Faiyaz Chaudhury	Backend Lead
Kaung Nay Htet (Kevin)	Backend Team
Poornank Purohit	Frontend Lead
Tolby Lam	Frontend Team, GitHub Master

Milestone 2

Milestone/Version	Date
M2V1	10/20/2022
M2V2	11/10/2022

Table of Contents

Back to Table of Contents

Back to Table of Contents

Section I: Data Definitions

1. **Administrative User:** a registered user with additional privileges to help keep the web application safe. Differentiated from a regular registered user by a leading “9” in their ID (which can be found in the database), and has a separate tab on the website’s navbar where admin privileges can be viewed.
2. **Bill:** an external transaction uploaded to a FunBill Transaction to be used in the application.
3. **Birthday List:** a list containing all the user’s friends, who have also toggled birthday lists on.
4. **Categories:** an attribute or tag given by the user to organize transactions
5. **Chat Box**: a social feature that allows registered users to communicate one on one with other registered users.
6. **Currency Type:** the type of currency being used in a transaction—must be a member of Currencies Supported
7. **Currencies Supported:** the total set of currencies supported by FunBill
8. **Currency Conversion:** a conversion between two different currency types
9. **FunBill Transaction:** a collection of Bills uploaded to FunBill with a total cost that is dynamically split between users or a group.
10. **FunBill Transaction Storage**: a transaction that has yet to be completed will be stored within the transaction storage.
11. **FunBill Transaction Log:** a log of all completed transactions
12. **General User:** a user that uses the web application without creating an account
13. **Groups**: a collection of users that will make up a group
14. **Home Feed:** a social feature that can post/display posts of registered users who use the application. It consists of registered users' friends’ posts and transactions
15. **Individual Receipt**: a copy of the Transaction Payment online.
16. **PIN:** a verification method to allow users certainty that they are paying the right person.
17. **Registered User:** a user that has created an account within the web application
18. **Reminders**: an alert to remind the receiving user that the transaction has been given and is due soon

Back to Table of Contents

19. **Scan Receipt**: scan receipt retains the data that was scanned from the physical copy of a receipt.
20. **Search**: users can use a search function to look up other users and/or past transactions
21. **Simplified Debt:** a currency value that is derived from a series of transactions between a group of two or more users.
22. **Transaction Payment**: an exchange of currency between two or more users in order to settle a FunBill transaction, typically of the amount either dictated by the transaction or the simplified debt algorithm.
23. **Transaction Payment Type**: each payment will have a method of payment, which will be one of the following: cash, Visa, Mastercard, American Express, PayPal, or CashApp.
24. **Transaction Pay Request:** a request sent from one user to another to pay an amount specified.
25. **Transaction Payment Request Description:** a Transaction pay request will have a Transaction pay request description to describe the reason for the payment request.

Back to Table of Contents

Section II: Prioritized Functional Requirements

Priority 1

1. General Users

1.1 A General user shall be an account

1.2 A general user shall be allowed to use the FunBill web application to participate in transactions

1.3 A general user shall have the option to sign up for an account and become a registered user

2. Registered Users

2.1 A registered user shall be an account

2.2 A registered user shall be able to search through their transactions

2.3 A registered user shall be able to search for other registered users

3. Groups

3.1 A group shall be comprised of one or more members

3.2 A registered user shall be able to create a group with other registered users

3.3 A registered user shall be able to add multiple transactions with multiple users

4. FunBill Transaction

4.1 A registered user shall be able to upload a transaction

4.2 A registered user shall be able to edit/delete an existing transaction until the transaction is complete/deleted

4.3 A registered user may be able to delete a transaction they requested

4.4 A transaction shall consist of at least one payment and at least two users, and a proportion that determines who owes how much

4.5 A registered user shall be able to change the transaction's total cost and proportions until the payment is made

4.6 A Pay Request may create a transaction

4.7 A registered user shall be able to invite other users to join a transaction

4.8 The user shall categorize transactions, but it is not mandatory

4.9 Registered users shall be given the option to charge another user or mark a payment as fulfilled to allow cash transactions

Back to Table of Contents

4.10 Any transaction edits shall be approved by all user's involved before edits take effect

5. FunBill Transaction Storage

5.1 Registered users shall be able to see their transaction storage and their current debt or money owed

6. FunBill Transaction Log

6.1 Transaction logs shall be an intractable table via category selection or through filtering

7. Individual Receipt

7.1 Individual Receipts shall be viewable once a transaction has been settled/completed

11. Transaction Payment

11.1 Transaction Payments shall be made using a single currency.

11.2 Transaction Payments will be processed immediately

11.3 Transaction Payments should have a payment type

11.4 Transaction Payment should be used to pay a Fun Bill Transaction

12. Payment Type

12.1 Payment Method can include cash

12.2 Payment Method can include Visa, Mastercard, or American Express cards

13. Pay Request

13.1 A pay request shall be used to create transaction

13.2 A registered user shall be able to send a Pay Request to another registered user

13.3 A pay request must have a value over 0.00 dollars

14. Transaction Payment Request Description

14.1 Registered users shall be able to decide if they want to have a transaction description describing the reason for the Pay Request

Back to Table of Contents

15. Simplified Debt

15.1 A registered user should be able to generate a simplified debt number

15.2 A registered user should be able to include any transaction within the collection

15.3 A registered user should be able to exclude any transaction within the collection

15.4 A registered user should be able to attach proof accompanying transactions within the group's simplified debt

16. Reminders

16.1 A registered user may add a reminder to a transaction for the receiving registered user

17. Search

17.1 A registered user shall be able to search for other registered users to add and send money to

17.2 A registered user shall be able to search for past transactions

17.3 A registered user shall be able to search by a date range

17.4 A registered user shall be able to search based on a specific category

17.5 A registered user shall be able to search based on keywords

18. Categories

18.1 Registered users shall be able to give each transaction a category

18.2 Registered users shall be given category options to choose from

18.3 Search shall be able to sort/filter results by category type

18.4 A Category should be defined to be either an Event, Outing, Shopping, Gas, Food, Rent, or Other

19. Home Feed

19.1 A registered user shall be able to post on the home feed

19.2 A home feed shall be toggled on or off by registered user

22. Chat

22.1 Registered users shall be able to chat with one another through the chat box

22.2 Registered users shall be able to interact with previous chats

Back to Table of Contents

23. PIN

23.1 A PIN shall be generated by a transaction between users who are not on each others' friends list

24. Admin

24.1 A admin shall be one account

24.2 An admin shall be able to delete posts from other users.

24.3 An admin shall have permission to edit/delete accounts

25. Singles

25.1 A singles shall be used to generate a payment request between two users.

25.2 A singles can be linked with a reminder

25.3 A singles can use a scan receipt

26. Account

26.1 An Account shall be able to create transactions with other users

26.2 An Account shall be able to create groups and group other users together

26.3 An Account shall be able to set a username

26.4 An Account shall be able to change their password at any time

26.5 An Account shall be able to invite others to a transaction

26.6 An Account shall be able to edit/delete a given transaction they are apart of

Back to Table of Contents

Priority 2

1. General Users

1.4 A general user shall not be able to change their username until they register for an account

2. Registered Users

2.4 A registered user shall be able to access the home feed

2.5 A registered user shall be able to post on the home feed

7. Transaction Receipt

7.2 Registered users can copy the transaction receipt and send them to other users

8. Currency Type

8.1 Registered users shall be able to choose their preferred currency type

8.2: A Currency type should be defined to be USD, Pound, Yen, Euro, other

9. Currency Supported

9.1 Registered users shall be given the option to choose a currency type among a currency supported

9.2 Payment currency shall consist of 10-20 international currencies

10. Currency Conversion

10.1 Registered users shall be able to convert the current transaction set currency type into a different currency type

10.2 If a currency conversion is requested by a registered user in a group, the other members of the transaction shall be notified before the change is made

10.3 Registered users shall be able to request a simplified debt after having derived from multiple costs that were present in multiple currencies

12. Payment Type

12.3 Payment Methods can include third party options like PayPal, CashApp

Back to Table of Contents

16. Reminders

16.2 A reminder will alert the receiving registered user at the time specified by the sending registered user

18. Categories

18.5 Registered users shall be able to make their own category

19. Home Feed

19.3 A registered user shall be able to comment on posts on their home feed

20. Scan Receipt

20.1 A registered user shall be able to upload an existing receipt in a group or a singles

20.2 A scanned receipt shall be used to generate a payment request

21. Treat List

21.1 A general or registered user shall have the option to be added to a list on the tab so that their expenses are covered by everyone else.

21.2 The treat list shall be toggled on or off by the registered user

26. Account

26.7 An account shall be able to change/edit their profile picture

26.8 An account shall be able to change/edit their password

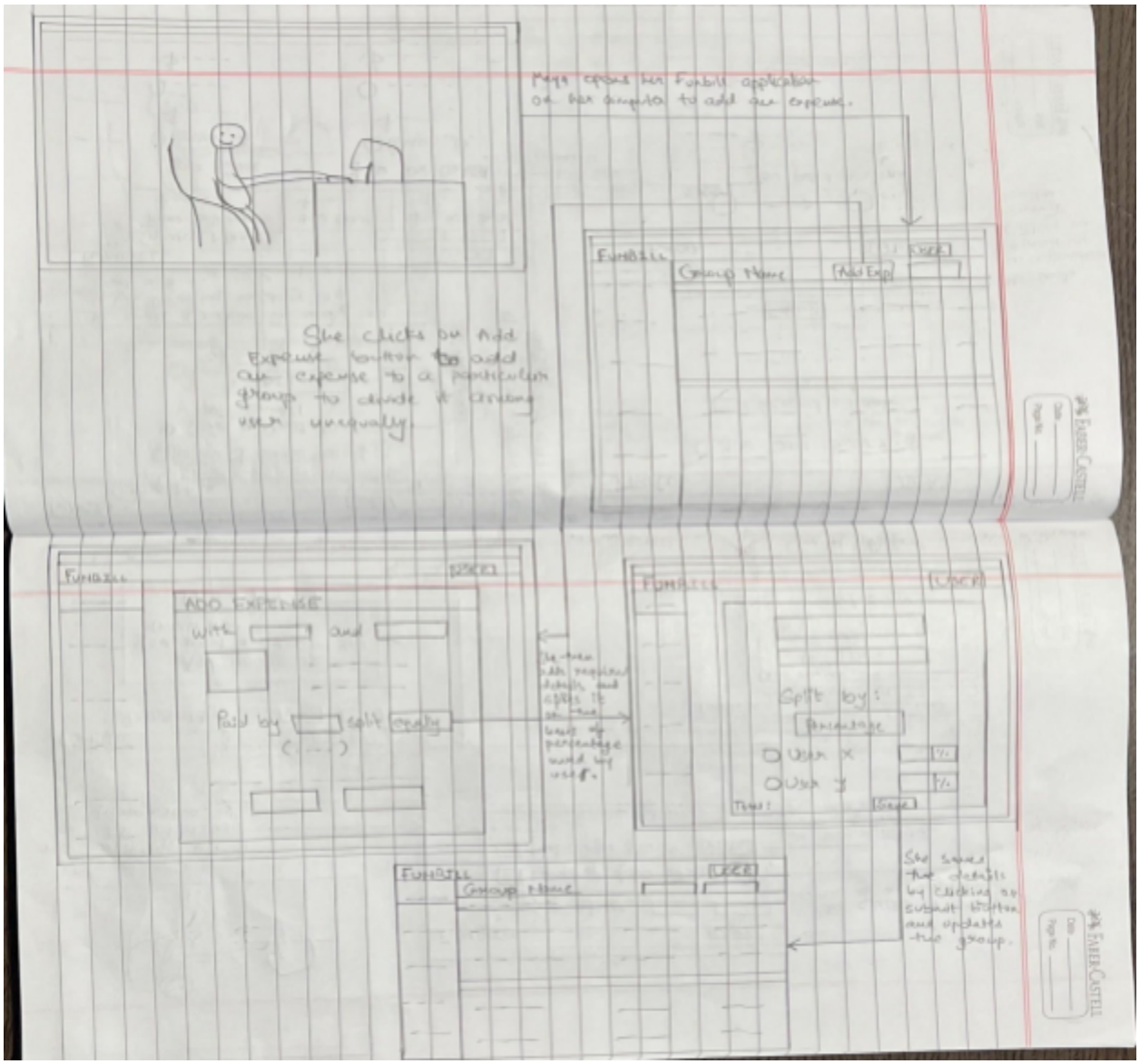
26.9 An account shall be able to change/edit their username

26.10 An account shall be able to update their emails

Back to Table of Contents

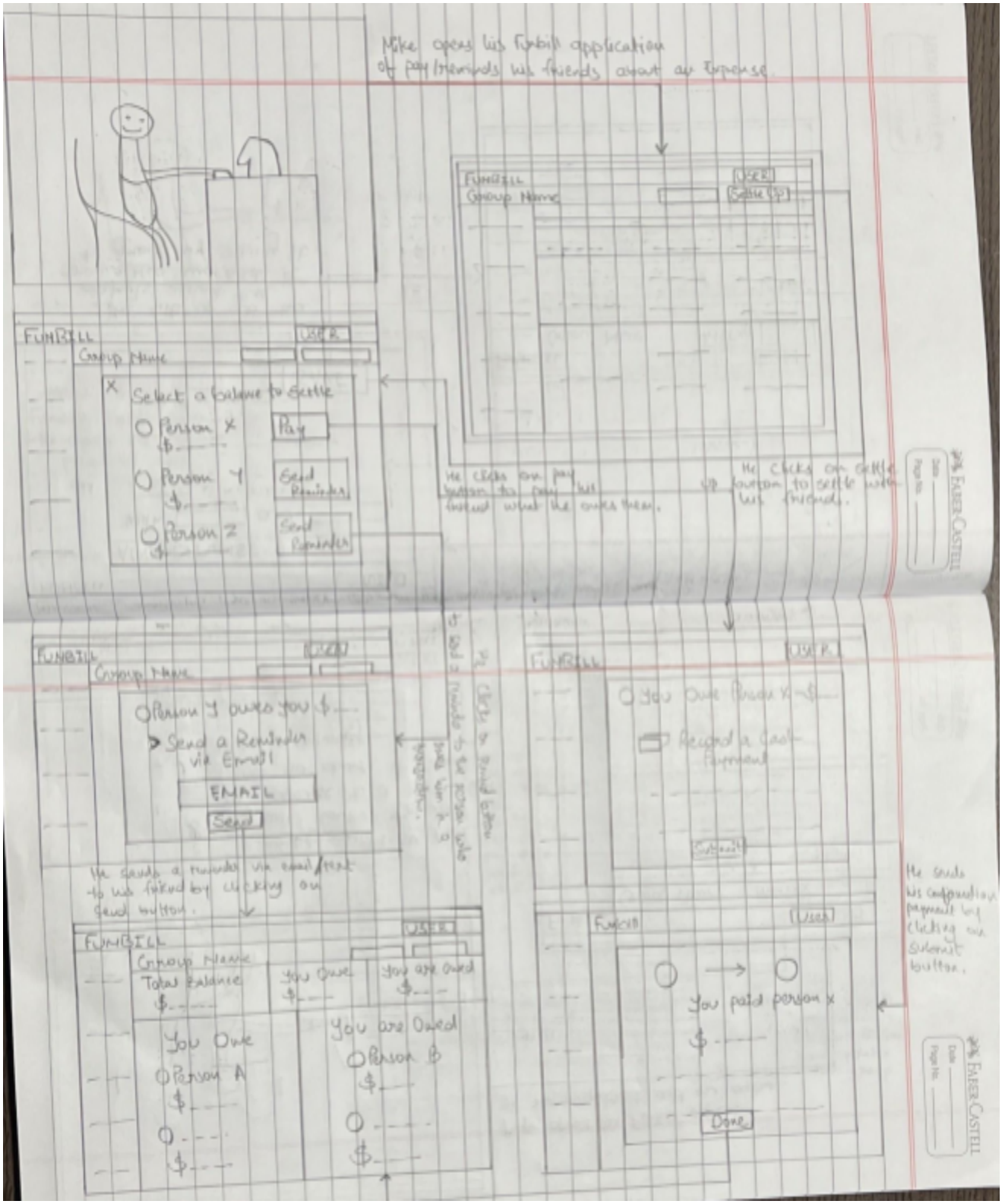
Section III: High-Level UI Mockups and Storyboards

1. Roommates want to split groceries



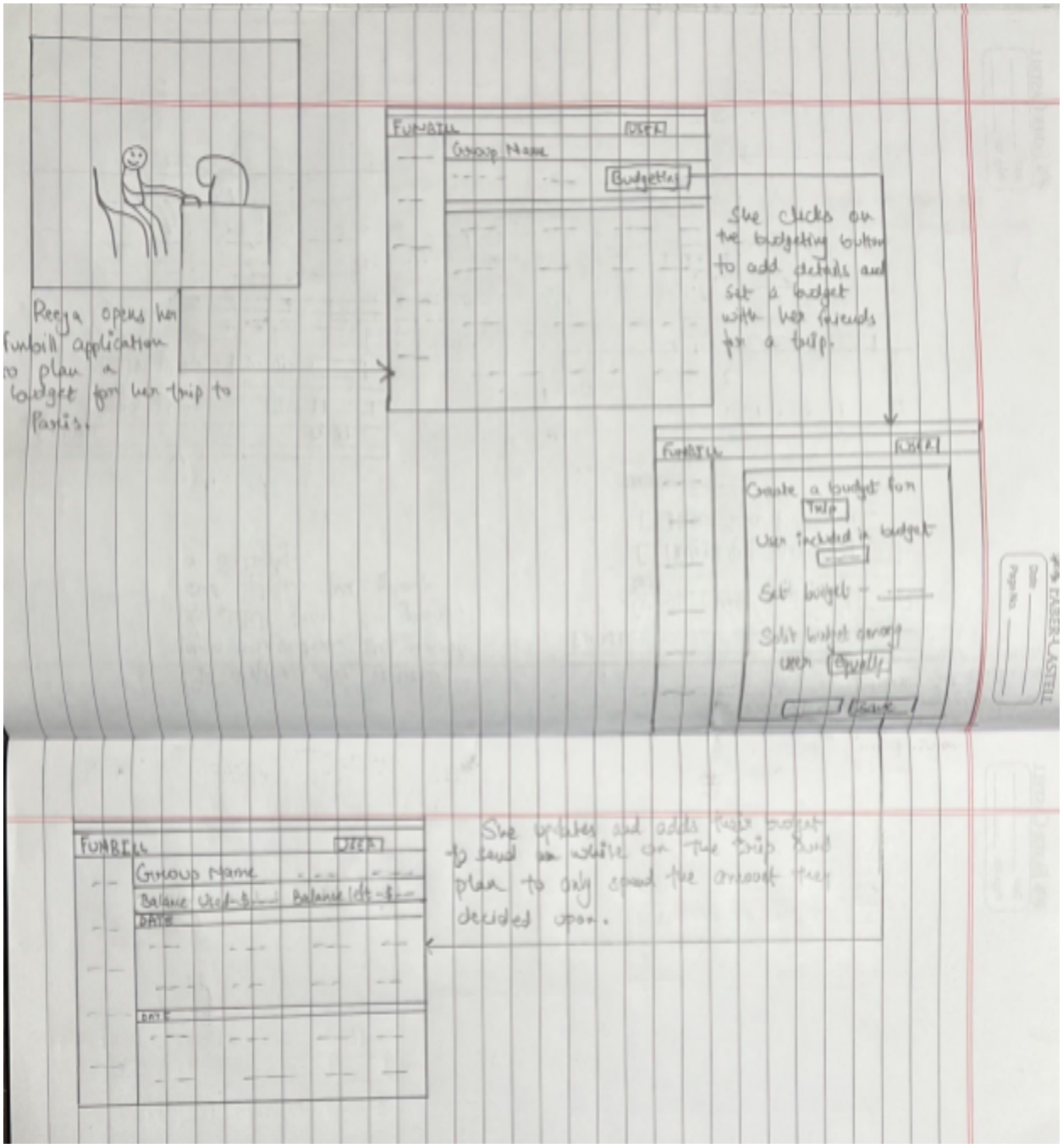
Back to Table of Contents

2. Payment Process and Reminders



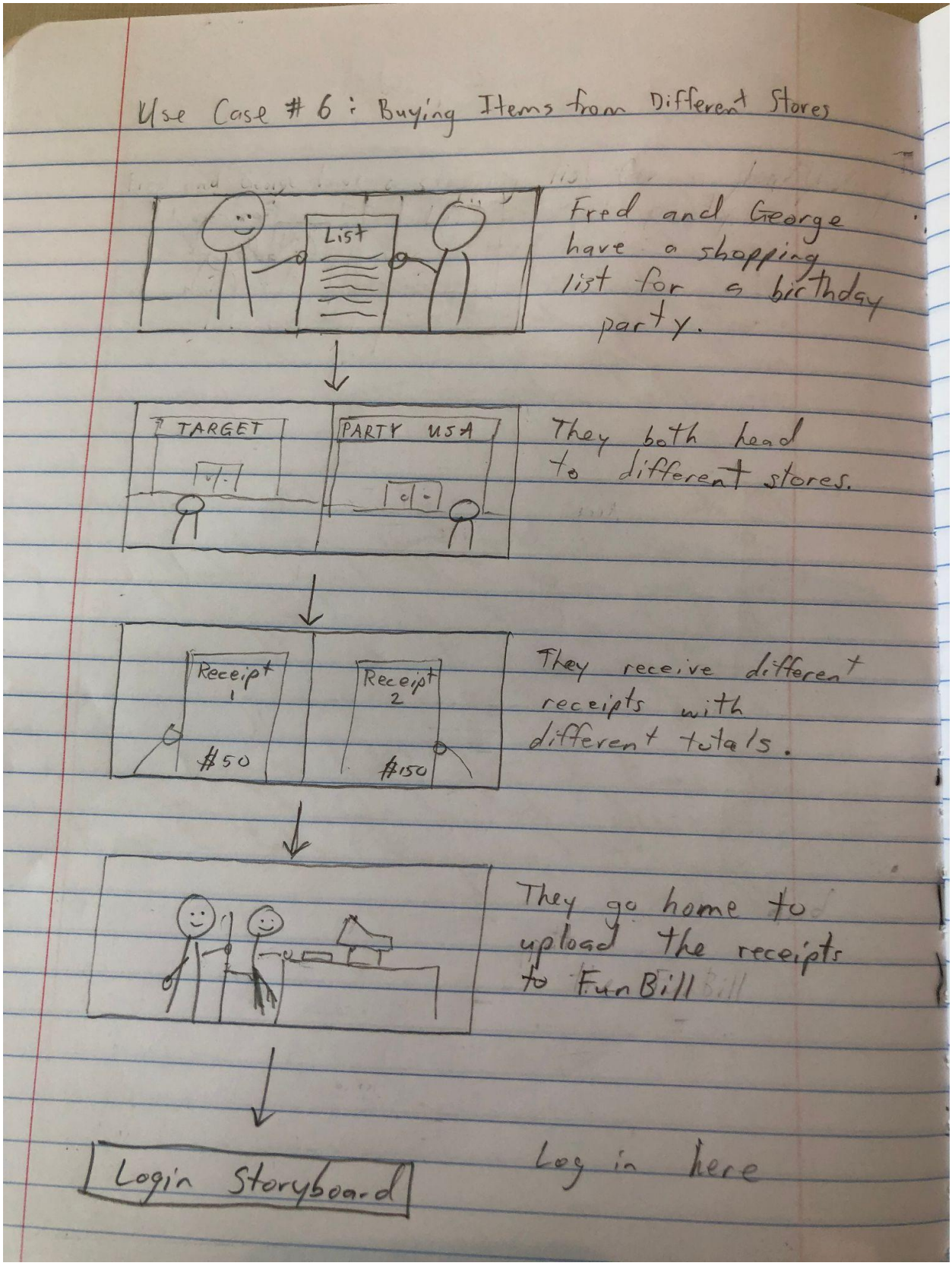
Back to Table of Contents

3. Budgeting on a trip with friends

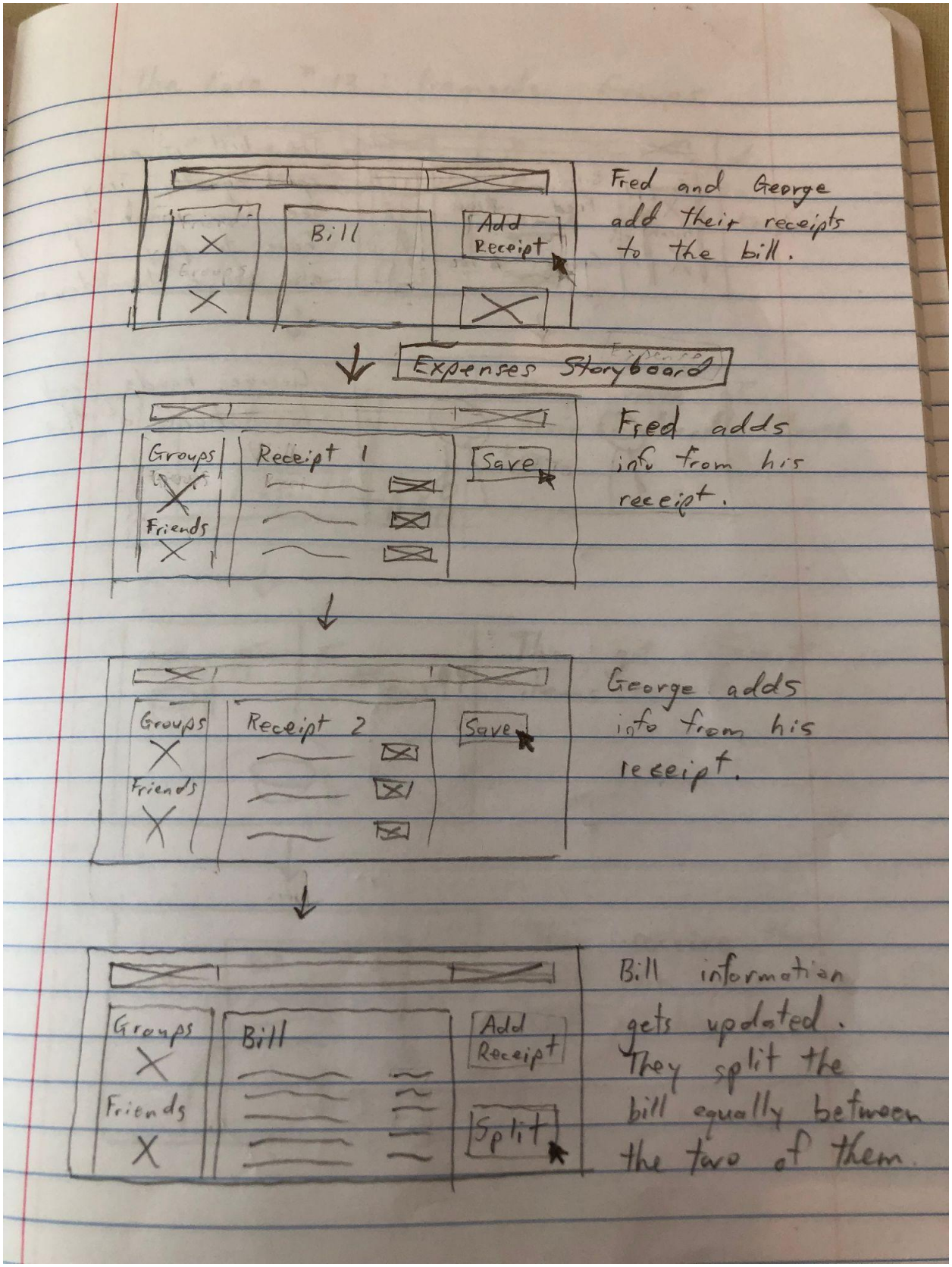


Back to Table of Contents

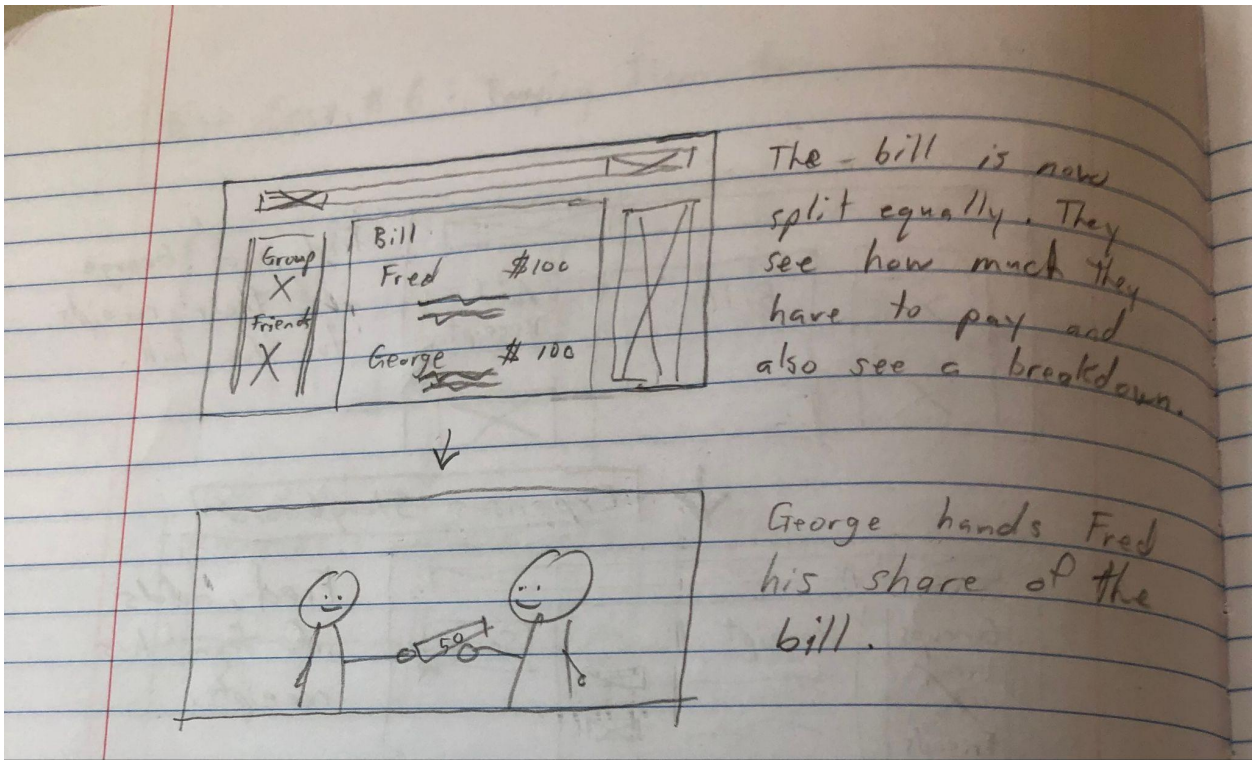
4. Buying items from different stores



Back to Table of Contents

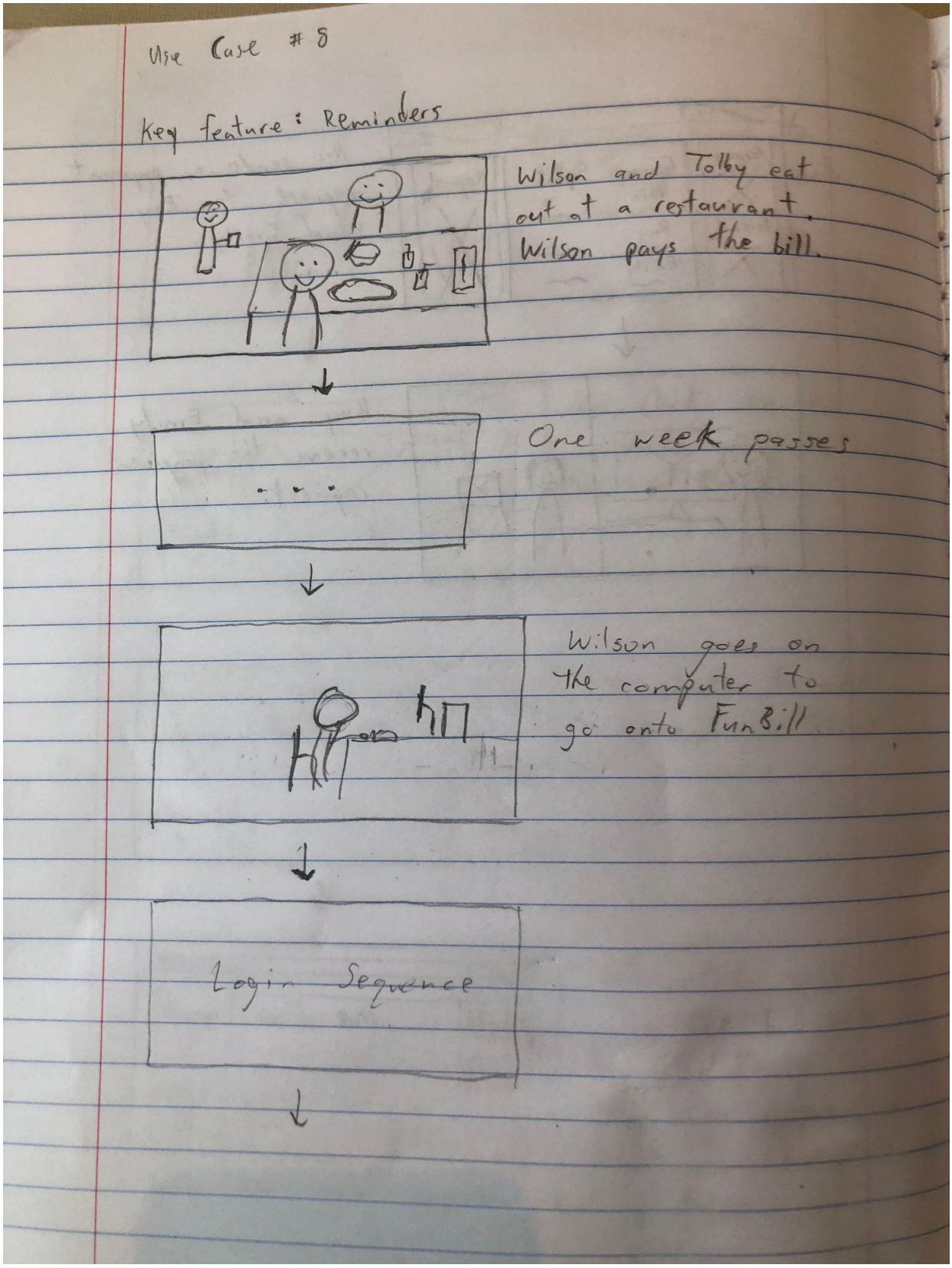


Back to Table of Contents

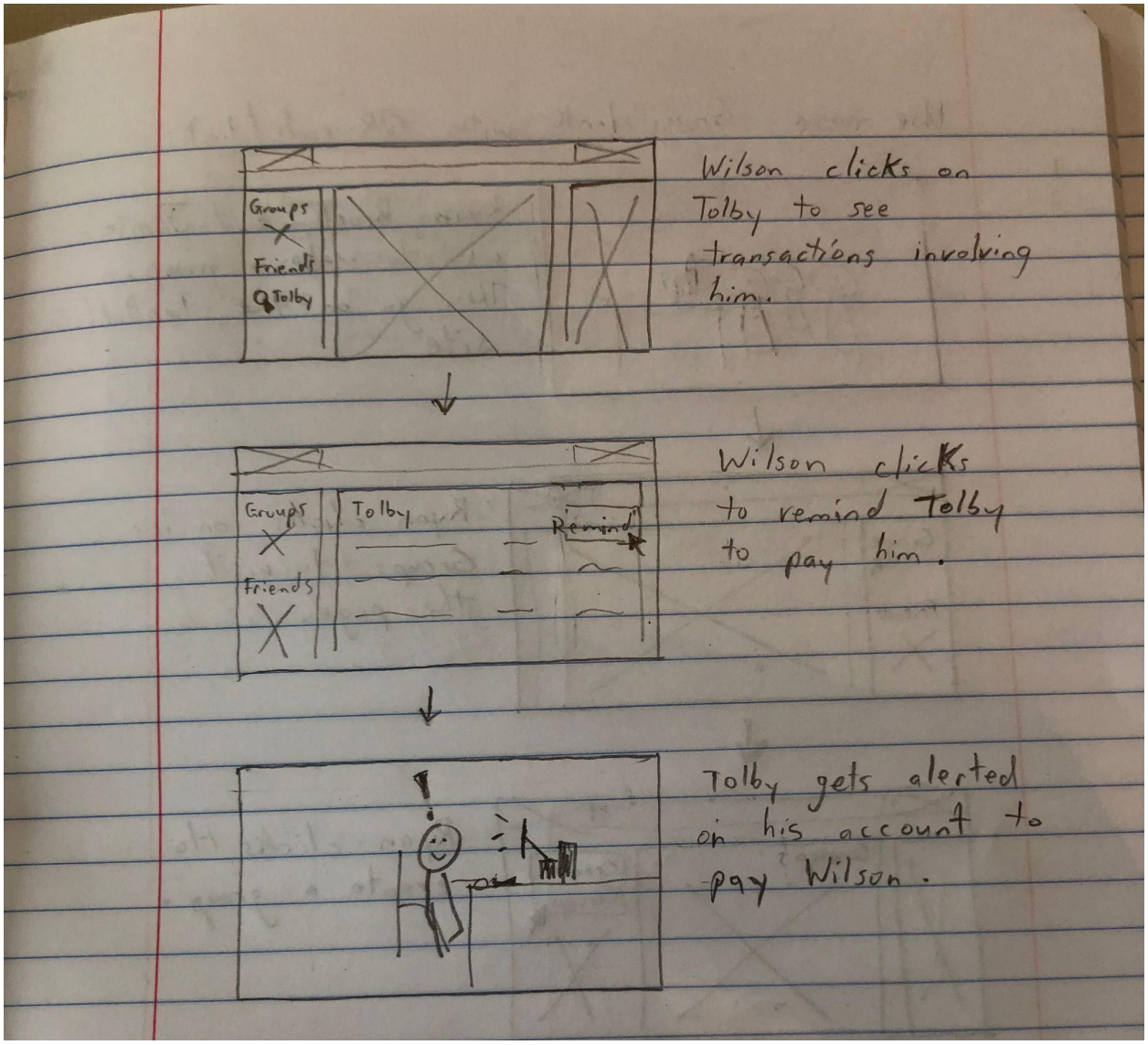


Back to Table of Contents

5. Reminders to a Forgetful Person

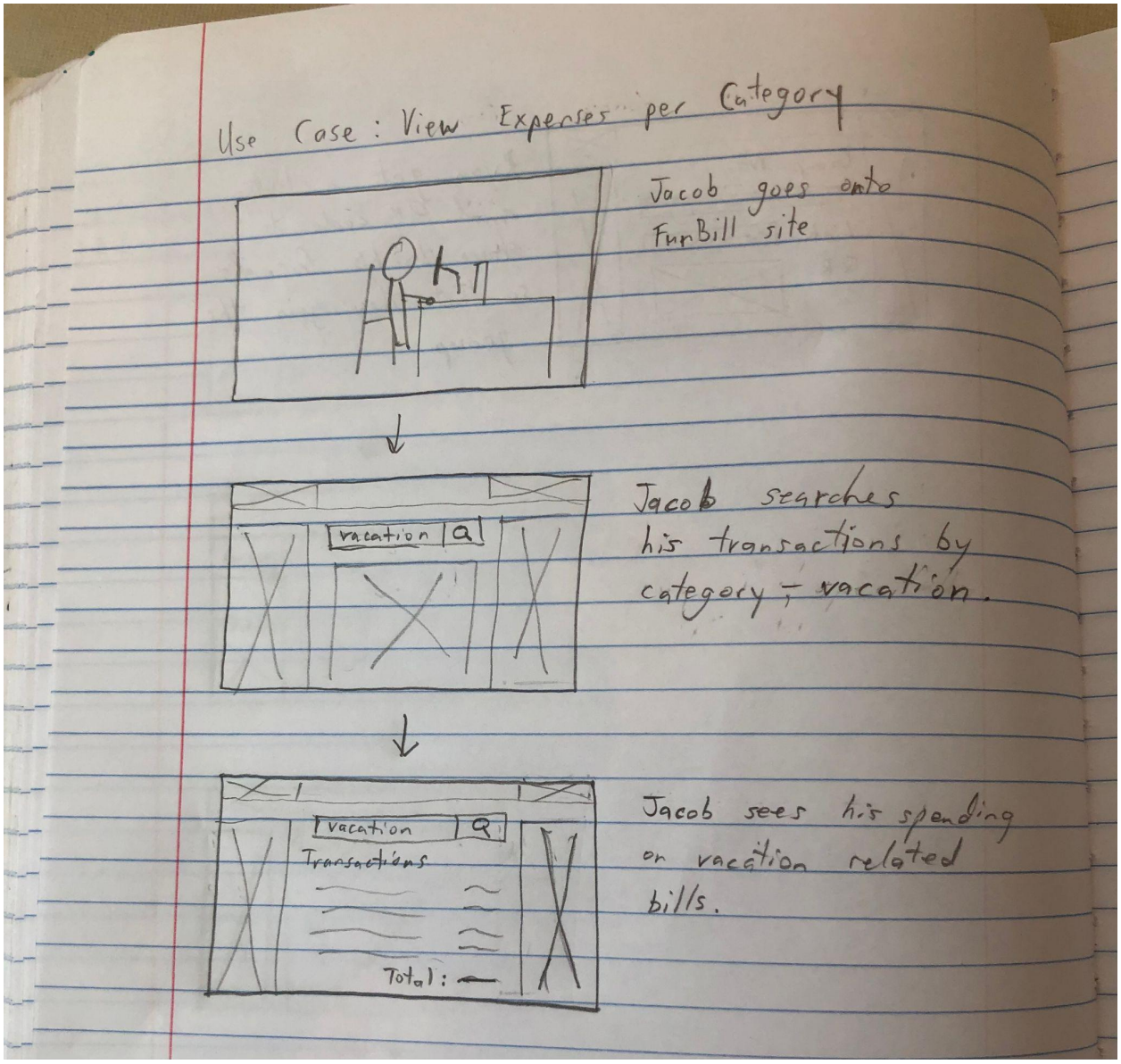


Back to Table of Contents



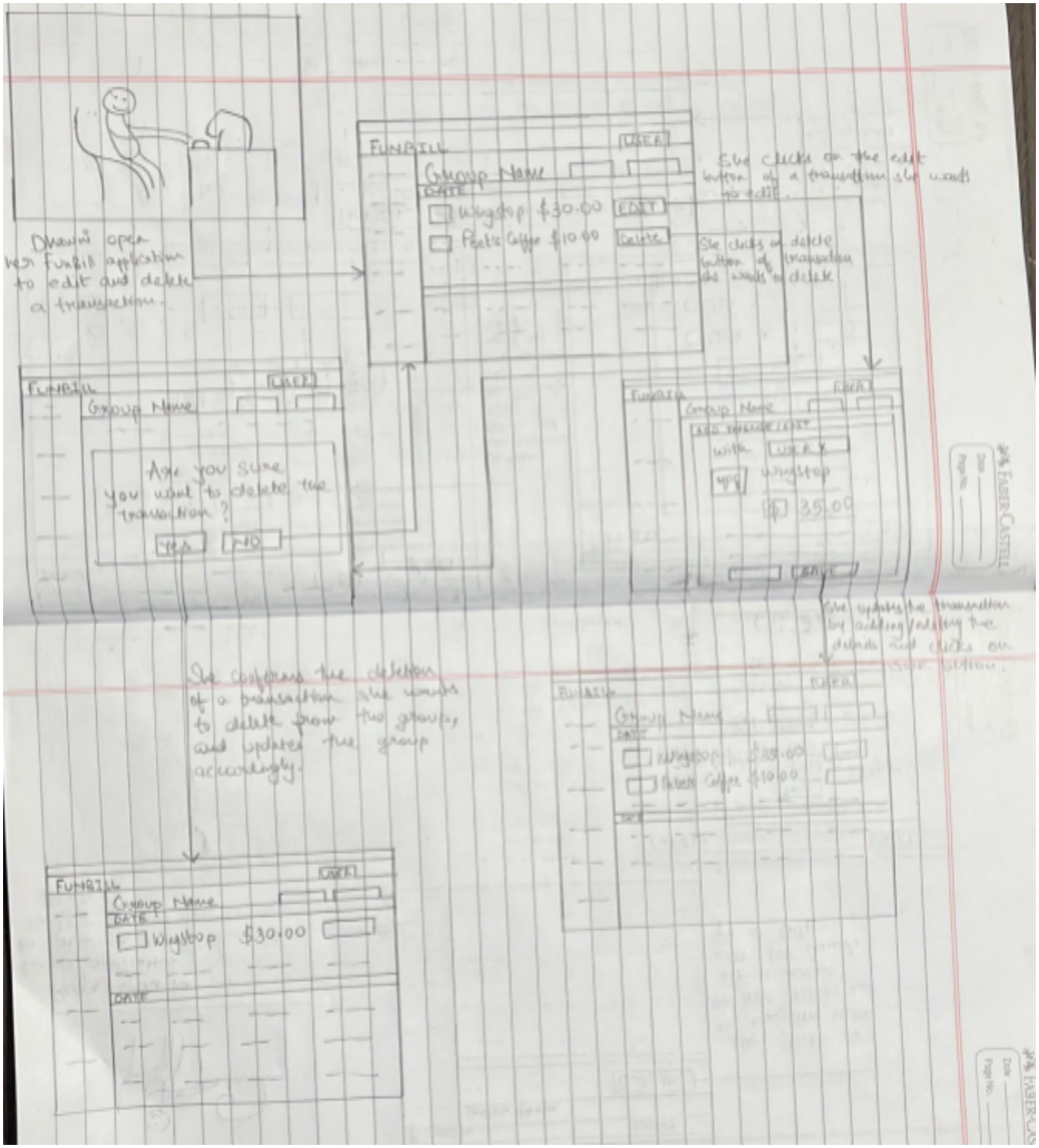
Back to Table of Contents

6. View expenses per category



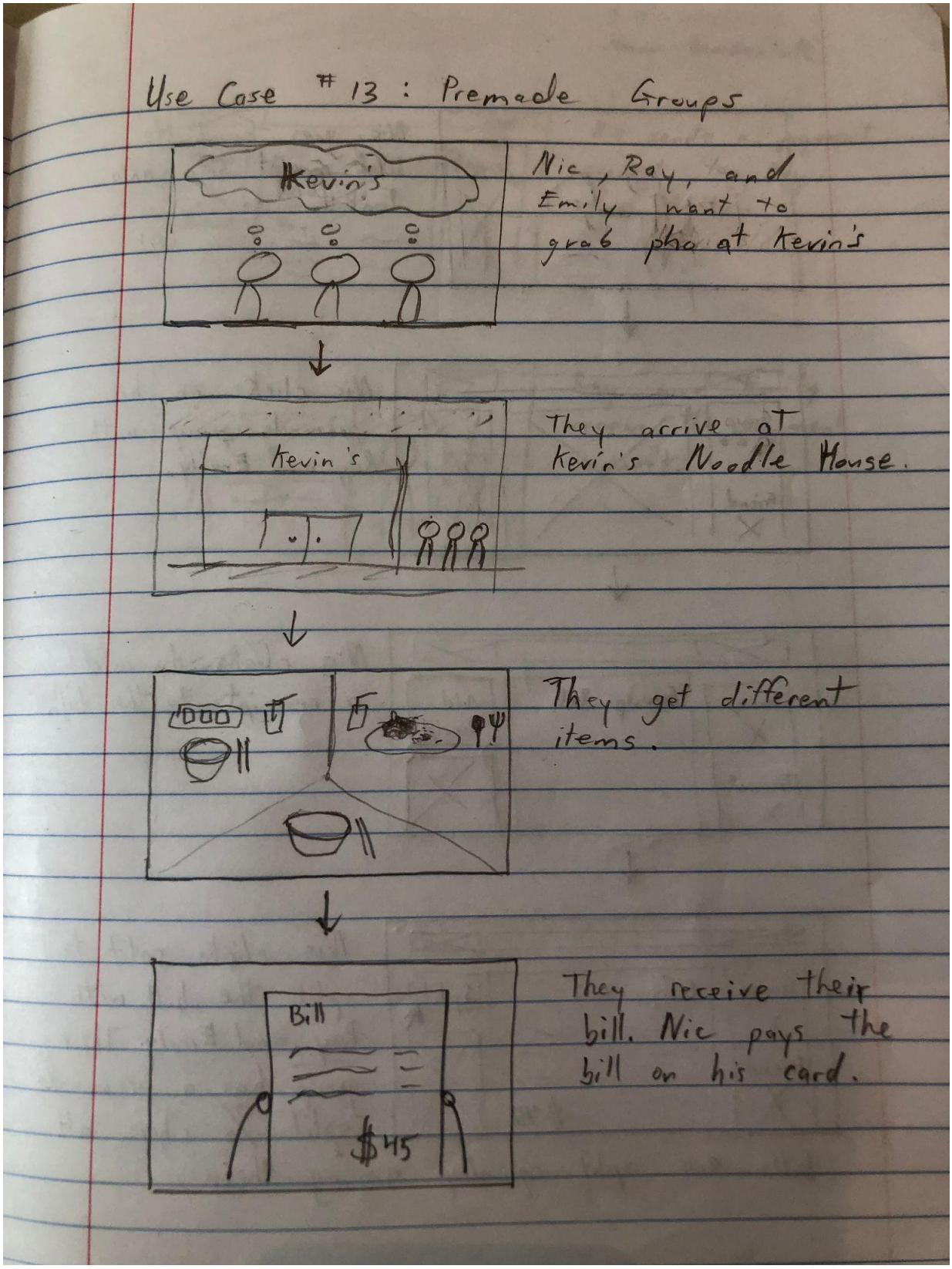
Back to Table of Contents

7. Editing and Deleting a Transaction

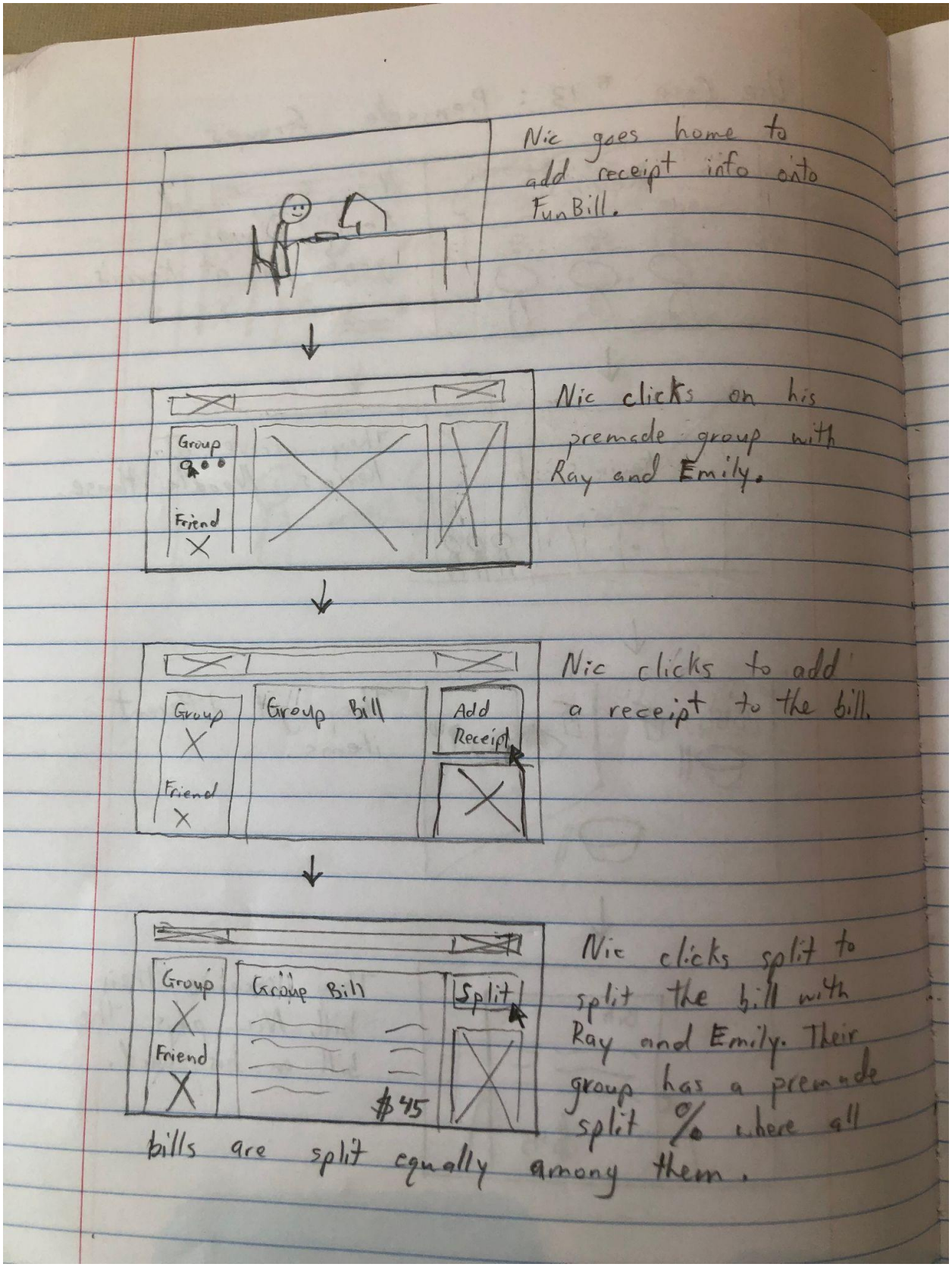


Back to Table of Contents

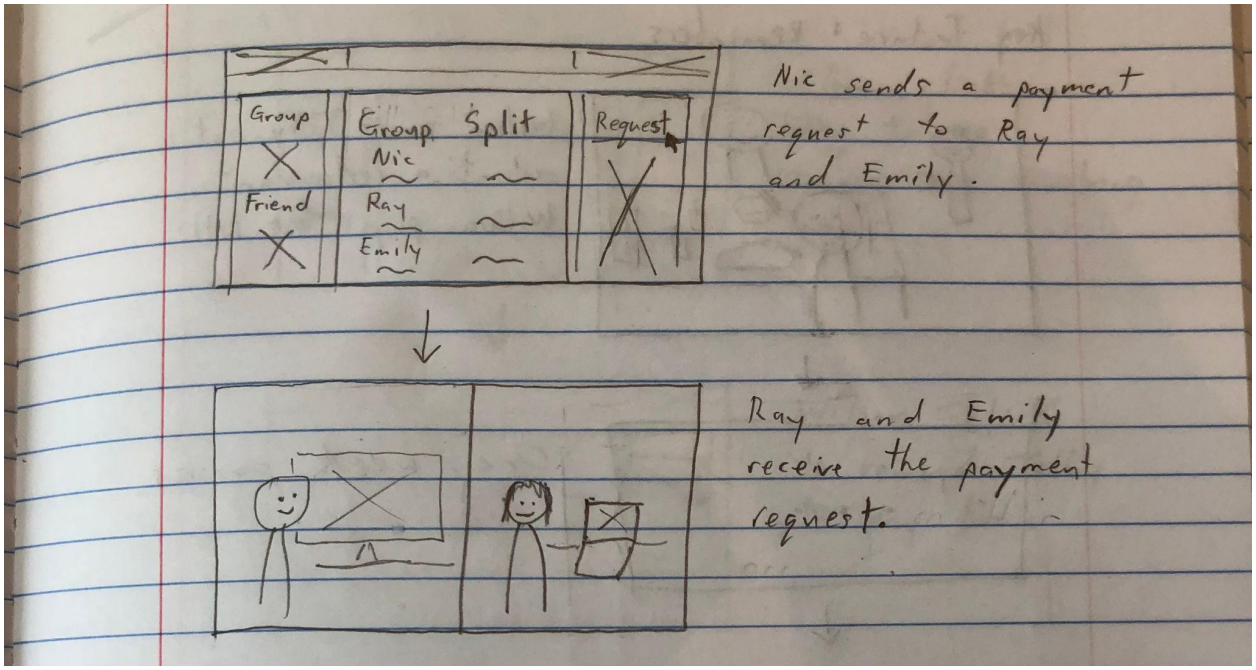
8. Premade Group



Back to Table of Contents

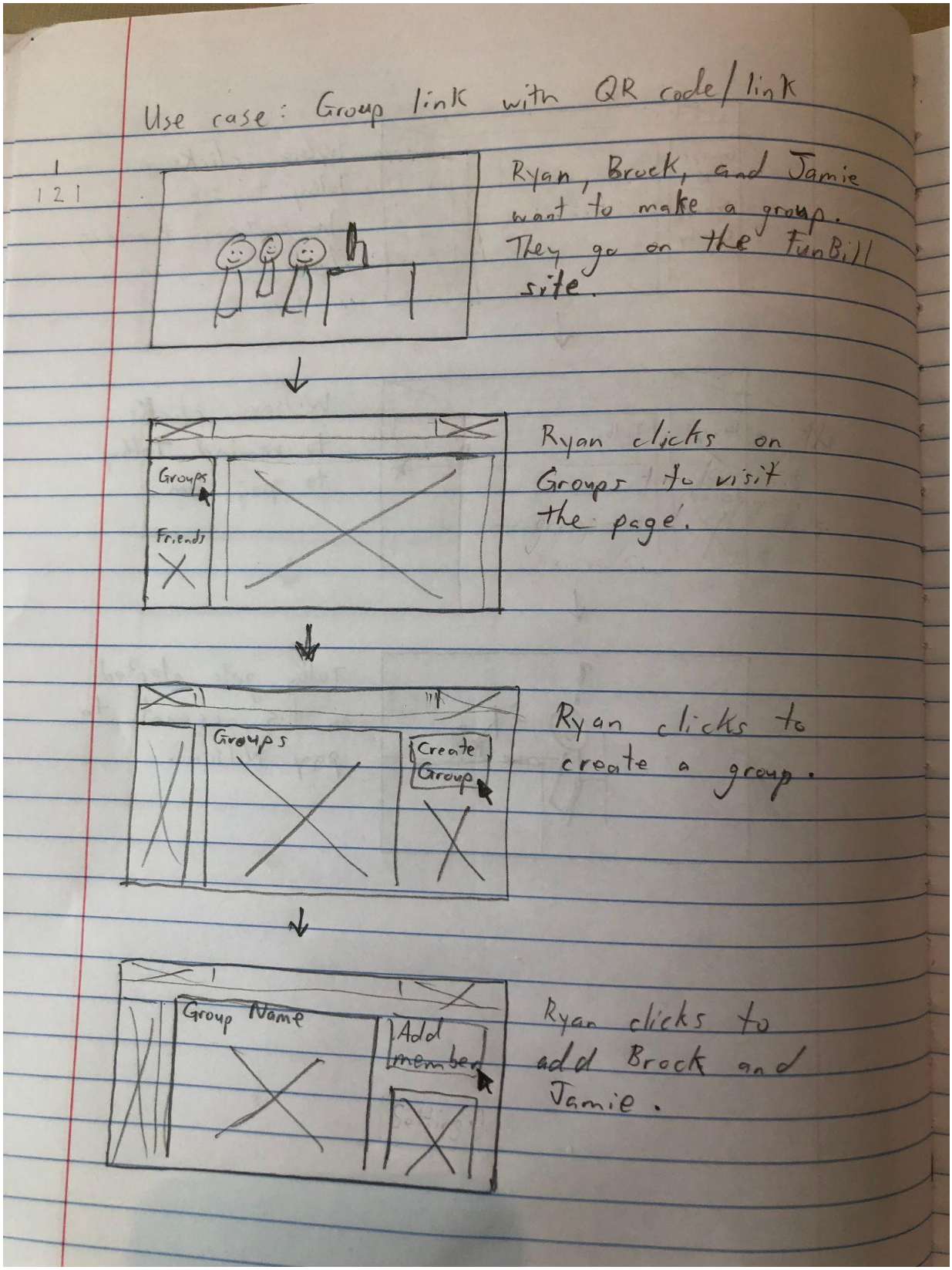


Back to Table of Contents

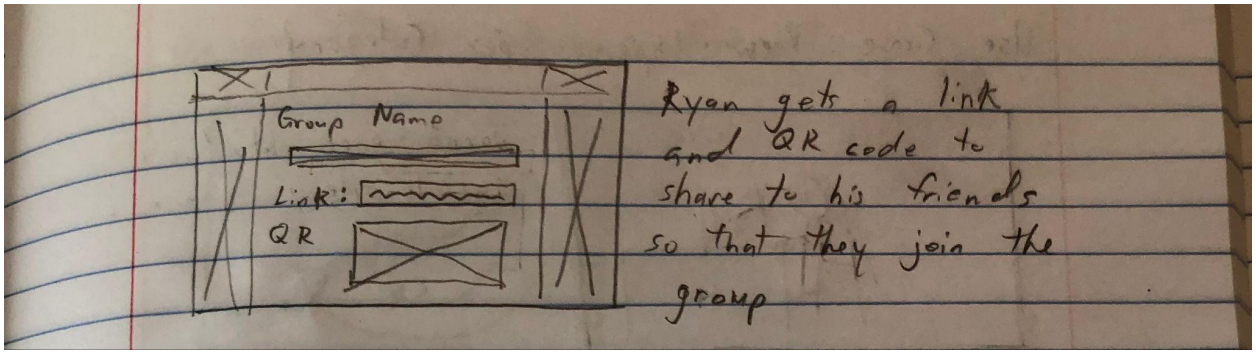


Back to Table of Contents

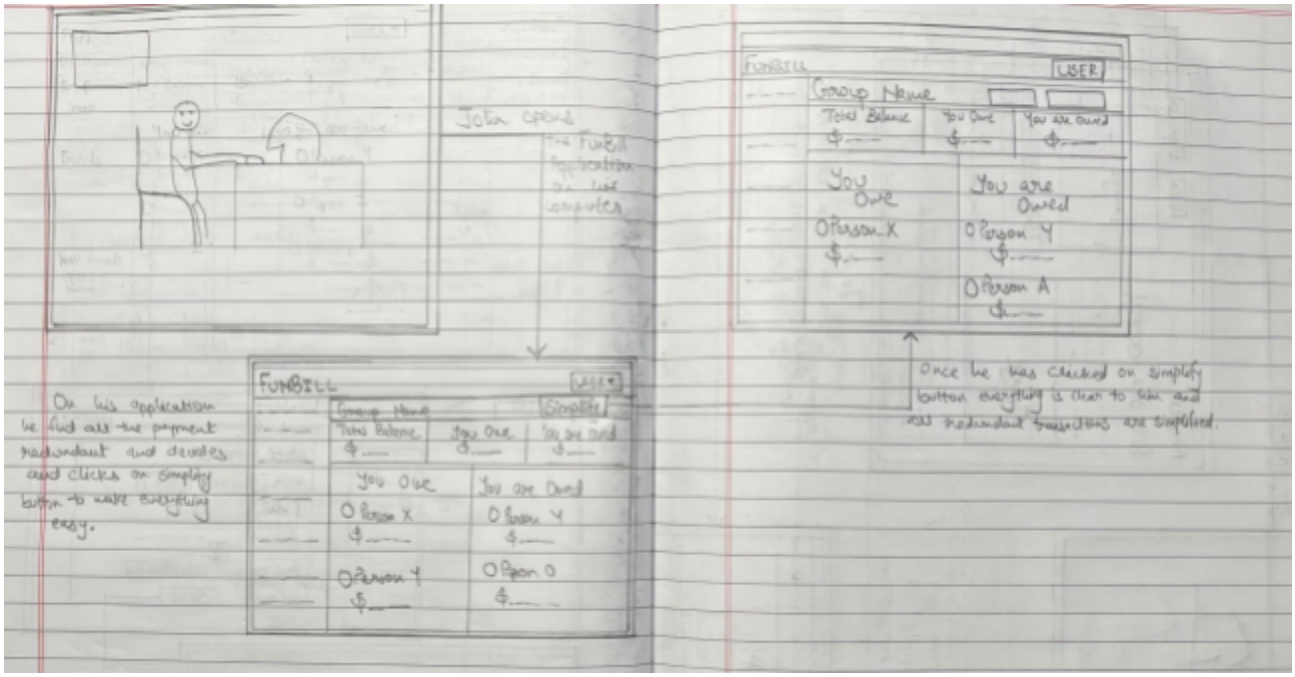
9. Group link with QR code/link



Back to Table of Contents

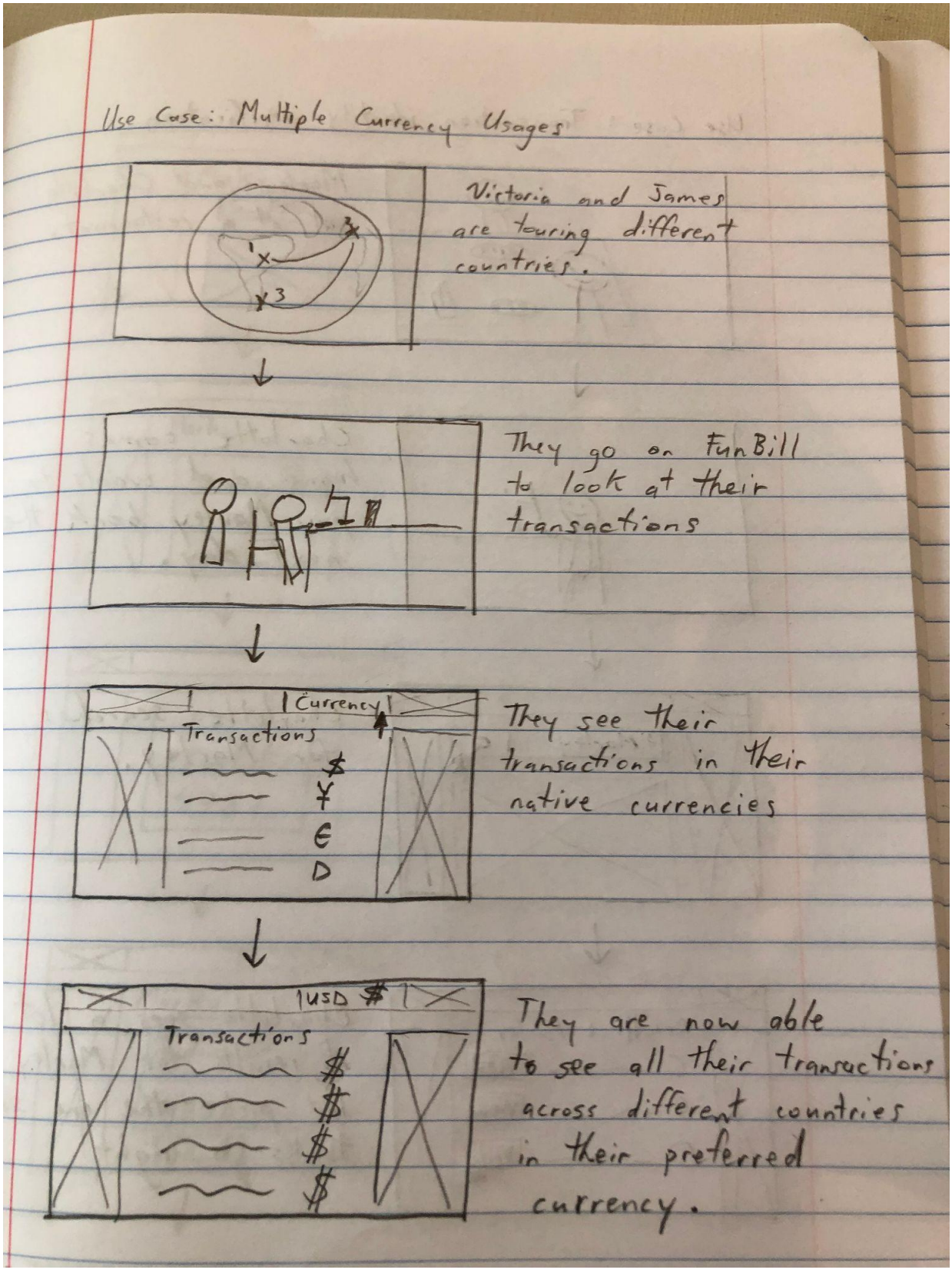


10. Simplify Group



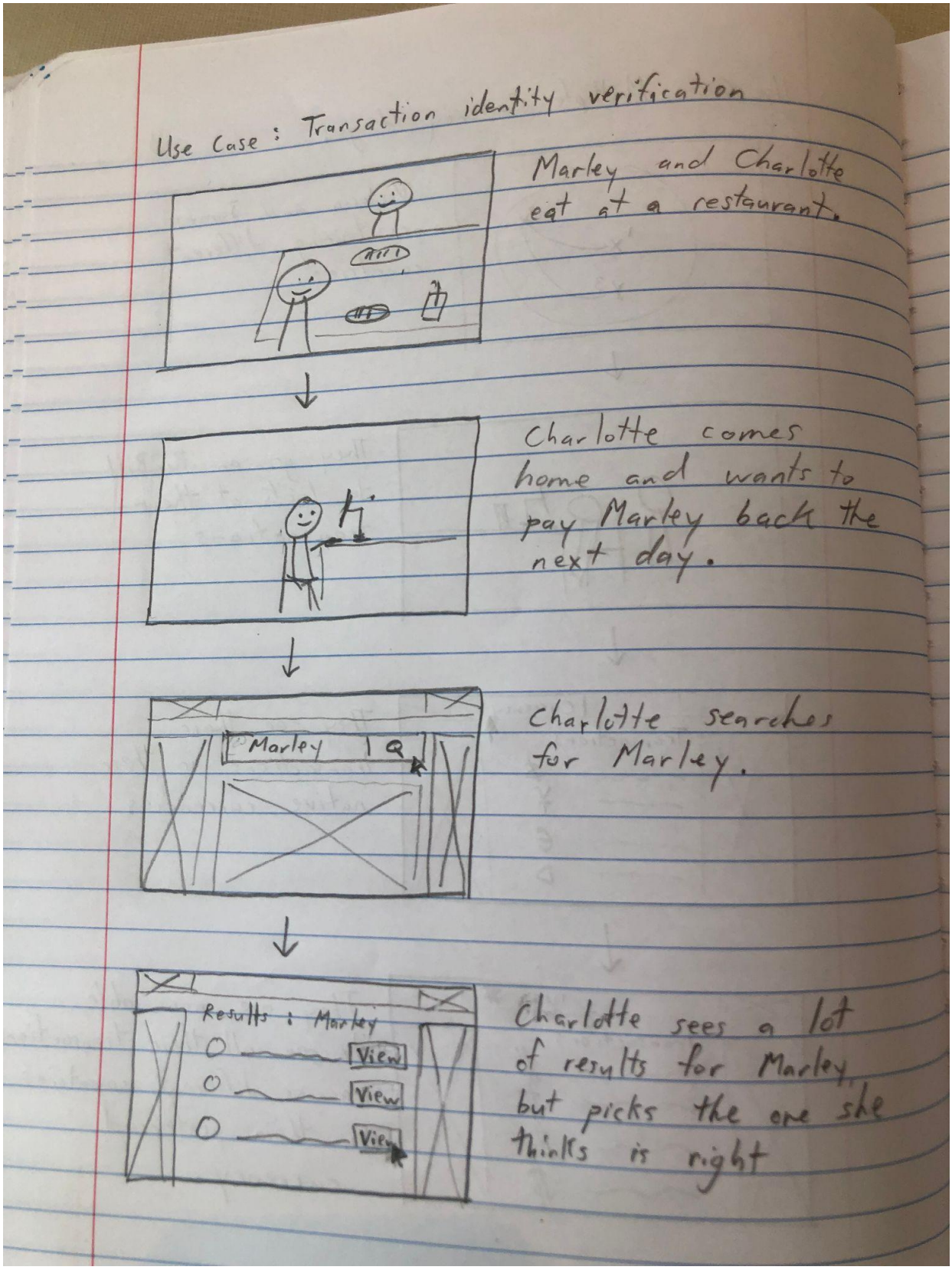
Back to Table of Contents

11. Multiple currency usages



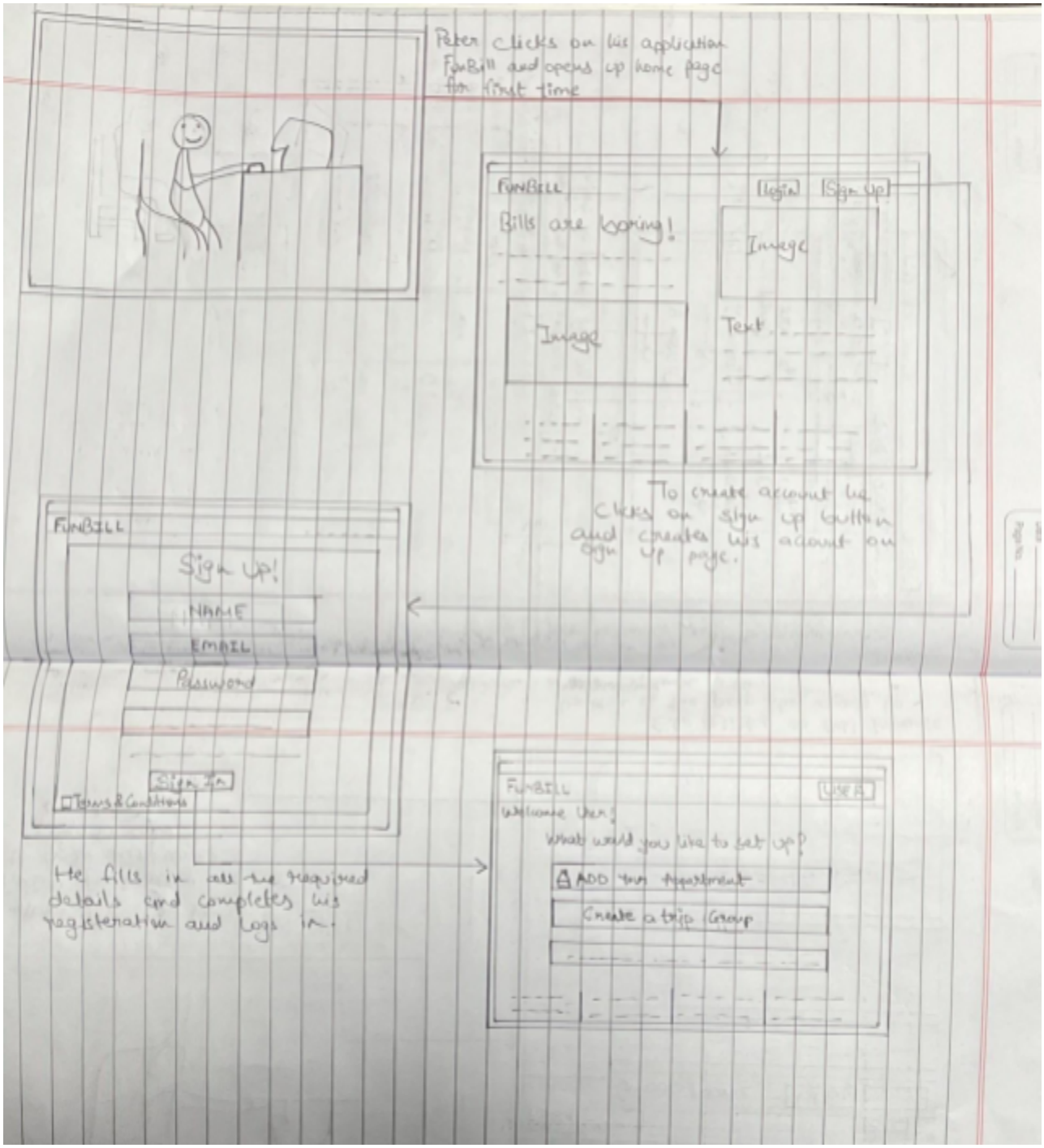
Back to Table of Contents

12. Transaction Identity Verification



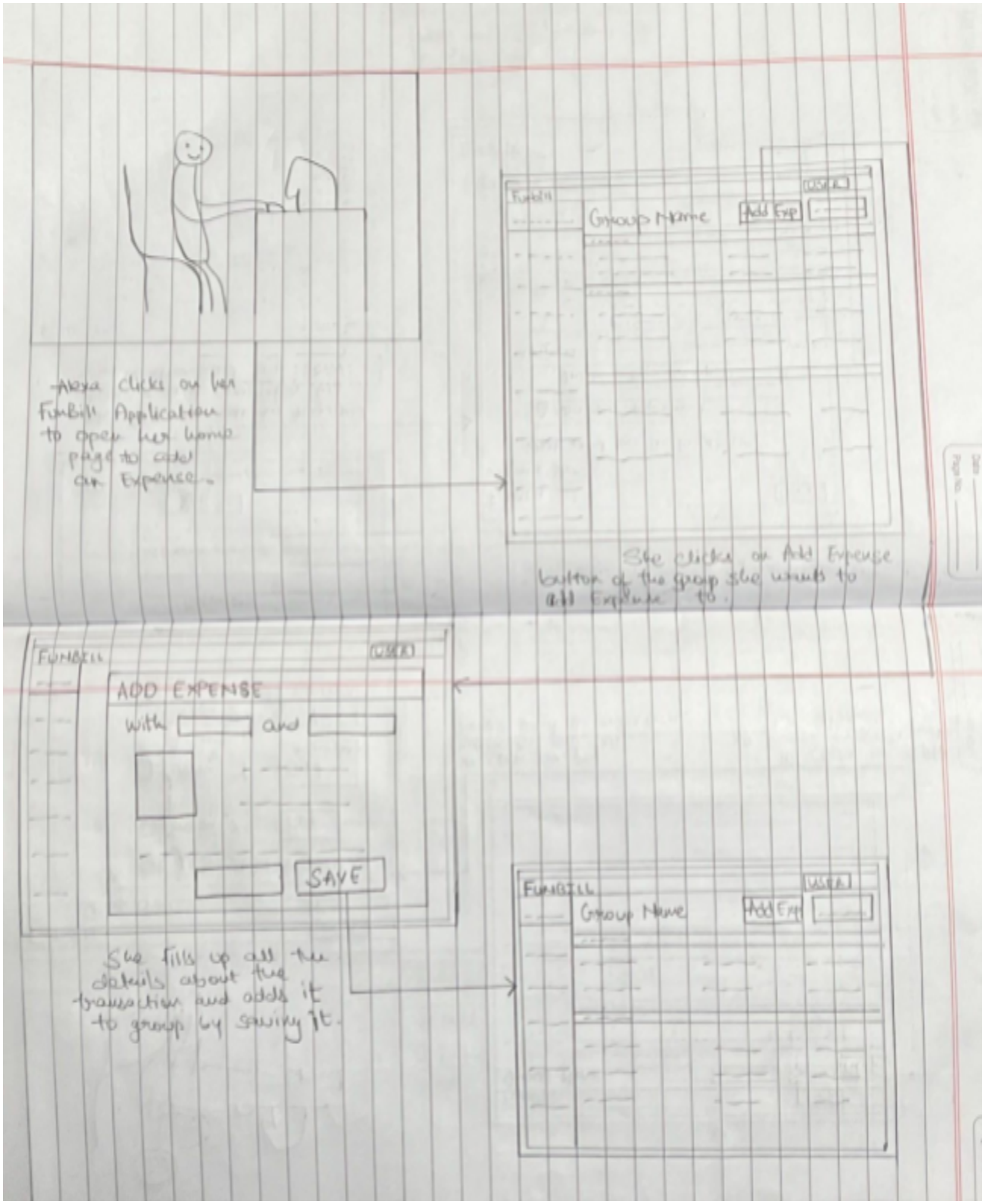
Back to Table of Contents

13. Signup and Registration



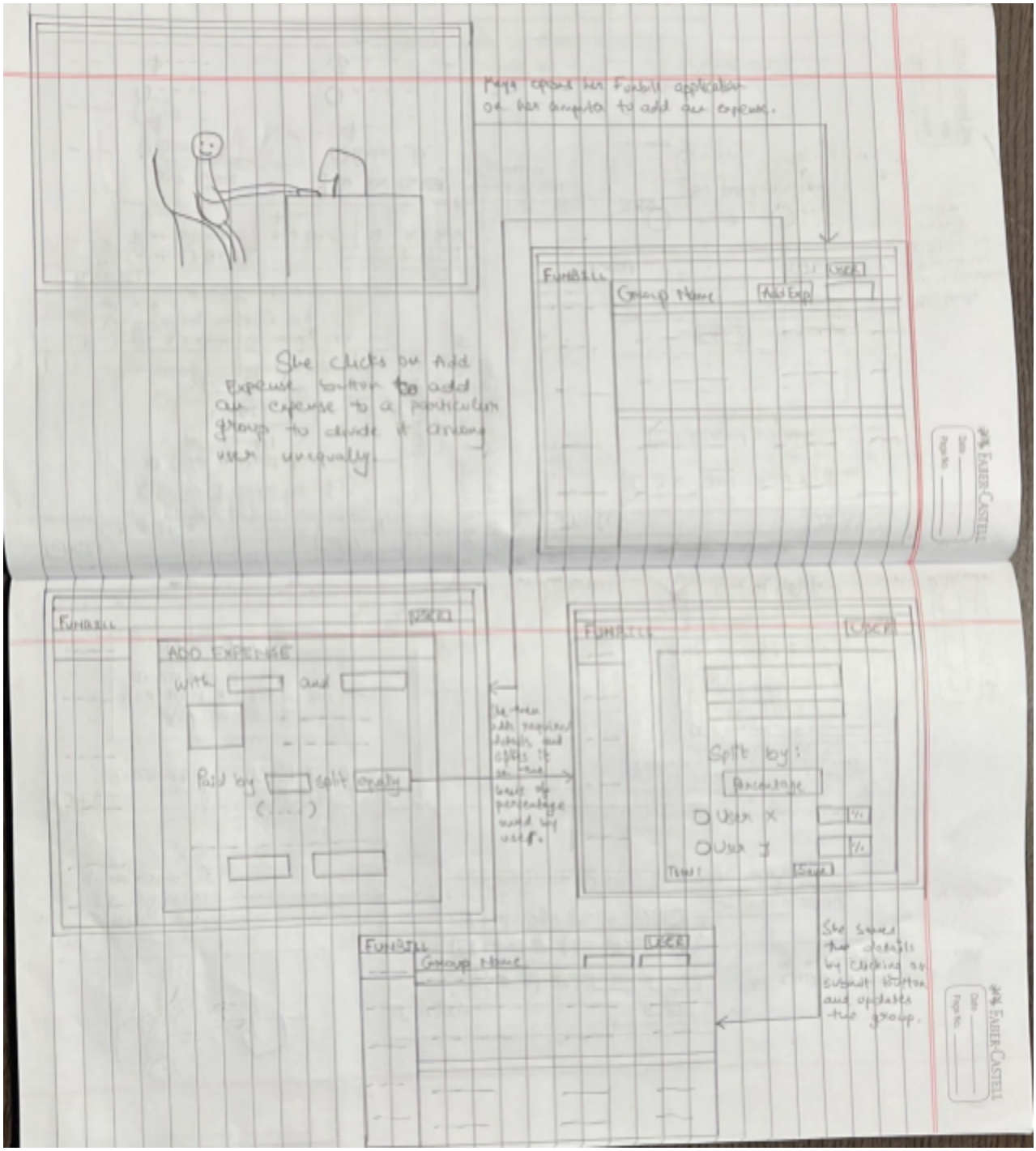
Back to Table of Contents

14. Adding an Expense



Back to Table of Contents

15. Splitting a Bill by percentage



Back to Table of Contents

Section IV: High-Level Database Architecture and Organization

Database Details

DBMS: MySQL
IDE: MySQL Workbench

Those of us who are familiar, with any sort of database work, have used MySQL as our DBMS. Half of the team is already familiar with MySQL, but those who need to learn will find MySQL to be one of the easiest to pick up. We are using MySQL Workbench as the IDE, as half the team is familiar with using it, so any confusion while using the IDE can be resolved by other team members.

Business Requirements

1. **Account:**
    a. An Account shall be defined to be a Registered User or an Administrator
    b. An Account can be managed by one or many Administrators
    c. An Account can be created from zero to one general user
    d. An Account shall be able to create zero to many groups
    e. An Account shall be able to create zero to many singles
    f. An Account shall be able to edit/delete many FunBill transactions they are a part of
    g. An Account must have zero to many payment methods
    h. An Account shall be able to create zero to many transactions
    i. An Account shall be able to create zero to many chats
    j. An Account shall hold a value for its username.
    k. An Account shall hold a value for their password
    l. An Account shall be able to set/reset the value of their username
    m. An Account user shall be able to change the value for their password
    n. An Account shall hold a value for its profile picture.
    o. An Account shall be able to change the value of its profile picture.
    p. An Account shall be able to invite one or many Accounts to a transaction
    q. An Account can delete many transactions they are linked with

Back to Table of Contents

r. Accounts shall be able to view zero to many transactions within transaction storage
s. Accounts shall be given the option to choose zero to one currency type among a currency supported
t. An Account shall be able to upload an existing receipt
u. An Account shall be given the permission to use zero to many simplified debts
v. An Account shall have the option to use zero to many Birthday Lists
w. An Account can view many FunBill Transaction Storage

2. **Administrators:**
    a. An Administrator is one and only one Account
    b. An Administrator is the one and only one user that can search for Accounts
    c. An Administrator is the one and only one user that can add new Accounts
    d. An Administrator is the one and only one user that can delete Accounts
    e. An Administrator is the one and only one user that can edit Accounts

3. **Birthday List:**
    a. A Birthday List shall have one to many accounts
    b. The Birthday list shall be given one transaction

4. **Categories:**
    a. A Categories is either an Outing, Event, Shopping, Gas, Food, or Rent
    b. A Categories can be linked to zero or many Transactions
    c. Categories can be used within zero to many Search

5. **Currency Type:**
    a. A Currency Type is defined to be USD, Pound, Yen, Canadian, Pesos, or other
    b. Currency types shall have zero to one registered users
    c. Currency Types shall contain a value
    d. Currency Types shall be linked to zero to one bill payments

6. **Currency Supported:**
    a. Currency supported has to exist on at least one transaction
    b. Currency supported shall have one or many currency types

Back to Table of Contents

7. **Chat:**
    a. Chat will be used to connect many Accounts

8. **General User:**
    a. A general user shall be allowed to participate in zero to many transactions
    b. A general user can create one account

9. **Groups:**
    a. A Group shall be comprised of zero to many accounts
    b. A Group shall be able to make zero to many Pay Requests
    c. A Group shall have zero to many Reminders
    d. A Group shall be able to have zero to many scan receipts

10. **Home Feed:**
    a. A Home Feed shall be linked to one registered user
    b. A home feed shall be toggled on or off by registered user

11. **Individual Receipt:**
    a. An Individual Receipt shall be generated from one and only one transaction payment

12. **Payment Method:**
    a. Accepted Payment Methods are Credit Cards and Debit Cards
    b. A Debit Card has one and only one Bank Account.
    c. Payment Method must have zero to many Registered Users
    d. Payments methods can have zero to many transactions

13. **Pay Request:**
    a. A Pay Request will be used to create one transaction
    b. A Pay Request can be created by one to many groups
    c. A Pay Request can be created by one to many singles
    d. A pay request must have a value over 0.00 dollars
    e. A Pay Request may contain an optional description

Back to Table of Contents

14. Registered User:

a. A Registered User is one and only one account
b. A Registered user shall be able to search for zero to many other registered users to add and send money to
c. A registered user shall be able to search for past transactions
d. A registered user shall be able to search by a date range
e. A registered user shall be able to search based on one and only one specific category
f. A registered user shall be able to post on zero to one home feed

15. Reminders:

a. A Reminder may be added to zero or one Transactions
b. A Reminder may be linked to one or many singles
c. A Reminder can be given to one to many groups

16. Scan Receipt:

a. A scanned receipt shall be linked to either one singles or group
b. A scanned receipt should be used to create a one-pay request

17. Search:

a. Search shall be able to look for many registered users
b. Search shall be able to look for zero to many transaction receipts of the searching registered user
c. Search shall have zero to many categories

18. Simplified Debt

a. Simplified debt should be able to see the data of one to many FunBill transactions
b. Simplified debt should hold a value of ignored transaction

19. Singles:

a. A Singles shall be made up of zero to many Accounts
b. A Singles can have zero to many pay requests

Back to Table of Contents

c. A Singles can have zero to many reminders
d. A Singles can have zero to one scan receipts

20. FunBill Transactions:

a. A FunBill Transactions shall have at zero to many Payment methods
b. A FunBill Transactions shall use zero to one currency selection
c. A FunBill Transactions can be created by zero to many accounts
d. A FunBill Transaction can be created by zero to many general users
e. A FunBill Transactions may contain zero to many reminders
f. A FunBill Transaction may be contained in zero to one Transaction Log
g. A FunBill Transaction shall have the option to have zero or one categories
h. A FunBill Transaction shall give data viewing permissions to zero to many simplified debts
i. A FunBill Transaction can have zero to many Birthday Lists
j. FunBill Transaction shall have one to zero Transaction Payment Request Description

21. FunBill Transaction Log:

a. A FunBill Transaction Log shall have one and only one Transaction
b. A FunBill Transaction Log shall be stored within one Transaction Storage

22. Transaction Payment:

a. A Transaction Payment shall contain one payment method
b. A Transaction Payment shall be linked to one Transaction
c. A Transaction Payment shall contain one Payment type.

23. FunBill Transaction Storage

a. FunBill Transaction Storage contains zero to many FunBill transaction logs
b. FunBill Transaction Storage shall be viewable to one to zero account
c. FunBill Transaction Storage shall give permissions to many searches

24. Transaction Payment Request Description

a. A Transaction Payment Request Description shall have one Funbill Transaction linked to it

Back to Table of Contents

Entities, Attributes, and Key Constraints

Entity	Attribute	Datatype	Key Constraint
Account	account_Id	int	PK
	email	varchar	PK
	userName	varchar	PK
	userPassword	varchar	—
	numTransactions	int	—
	birthday	date	—
Administrators	admin_id	int	PK, FK
	Account	int	PK
Birthday List	birthdayList_id	int	PK
	Account	int	—
	FunBillTransaction	int	FK
	birthDate	date	—
Categories	categories_id	int	PK
	category	enum	—
Chat	chat_id	int	PK
	numNewMessage	int	—
Currency Supported	currenciesSupported_id	int	PK
	CurrencyType	int	FK
Currency Type	currencyType_id	int	PK
	type	enum	—
General User	generalUser_id	int	PK, FK
Groups	Groups_id	int	PK
	GroupTitle	varchar	—
Home Feed	homeFeed_id	int	PK
	RegisteredUser	int	FK
Individual Receipt	individualReceipt_id	int	PK
	datePaid	date	—
	totalPaid	int	—
	TransactionPayment	int	FK
Pay Request	payRequest_id	int	PK
	payAmount	int	—
	FunbillTransaction	int	FK

Back to Table of Contents

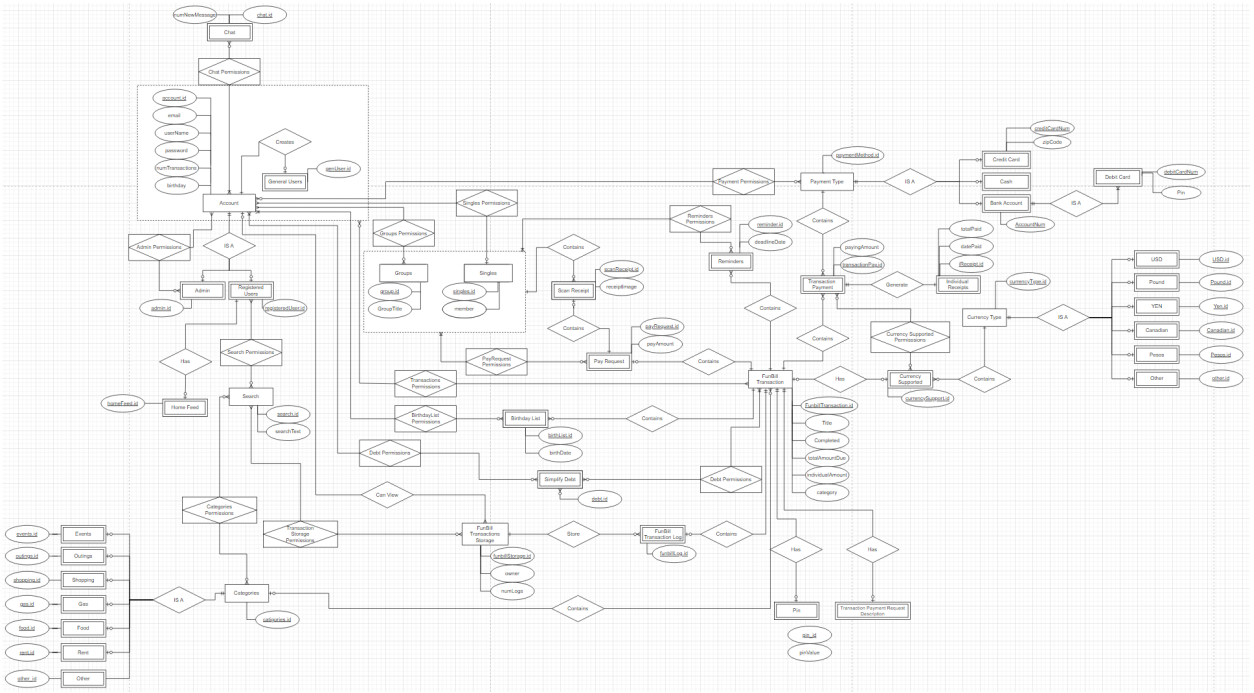
Entity	Attribute	Datatype	Key Constraint
Registered User	registeredUser_id	int	PK
	Account	int	PK, FK
	registeredUsercol	int	—
Reminders	Reminders_id	int	PK
	FunbillTransaction	int	FK
	deadlineDate	date	—
Scan Receipt	scanReceipt_id	int	PK
	PayRequest	int	FK
	receiptImageLink	varchar	—
Search	search_id	int	PK
	searchText	varchar	—
Singles	Singles_id	int	PK
	member	int	—
FunBill Transactions	funBillTransaction_id	int	PK
	Category	int	FK
	CurencySupported	int	FK
	Title	varchar	—
	Completed	boolean	—
	totalAmountDue	int	—
	individualAmount	int	—
FunBill Transaction Log	funBillTransaction_id	int	PK
	funBillTransactionStorage	int	PK, FK
	FunBillTransaction	int	PK, FK
Transaction Payment	transactionPayment_id	int	PK
	PaymentType	int	PK, FK
	FunBillTransaction	int	PK, FK
	payingAmount	int	—
FunBill Transaction Storage	funbillTransactionStorage_id	int	PK
	owner	int	FK
	numLogs	int	—
Transaction Payment Request Description	transactionDescription_id	int	PK
	FunBillTransaction	int	PK, FK
	description	varchar	—

Back to Table of Contents

FunBill ERD

This diagram represents the entire ERD for the FunBill web application.

Link to ERD: [FunBill ERD]



Media Storage

Resource used: [Should binary files be stored in the database?]

1. Storing media files in a database with a blob
    - Con: files can be unnecessarily large, making it hard to back up, process, and scale up
2. Storing media files on a filesystem with a link in the database
    - Con: easily breakable with a large number of links within the database.
    - Con: moving from one server to another may be tricky as ownership permissions are not as easily transferable
    - Con: hard to keep the links unscalable, due to the 256-character limit on a typical file system
3. Storing media files on a filesystem and renaming to a hash of contents and storing the hash on the database

Back to Table of Contents

Search/Filter Architecture and Implementation

Search functionality takes several components and has its own functional and non-functional requirements.

- Functional Requirements
    - All entries relevant to a keyword entered by the user must be found and returned from the database.
    - The entries must be arranged in order according to relevance to the keyword and popularity.
- Nonfunctional Requirements
    - Low latency—Results must appear as quickly as possible.
    - High throughput—The engine must be able to respond to multiple queries simultaneously.
    - Scalability—The system should be accommodating for a growing number of users and entries.
- Components
    - Crawler
        - The purpose of the crawler is to fetch all the entries. This is more necessary on web browsers, but if we search through all entries looking for certain phrases/words it should be analogous.
    - Indexer
        - The indexer indexes entries cached by the crawler. This makes lookup faster since we can use a unique index to locate specific entries.
        - Database uses LIKE…WHERE
        - Stem words included in transaction descriptions/any information put in by the user (date, price, users, location?, store info if using receipt scanner)
        - Inverting the index makes lookup faster (keyword -> index -> documents)
    - Retriever
        - The retriever uses the index created by the indexer to answer users' queries.

Resources Used: System Design Interview: Search Engine | Tech Wrench, How to create a search function for your website using Javascript, PHP and MySQL

Back to Table of Contents

Section V: High-Level APIs and Main Algorithms

Dwolla API

Dwolla can be used for the payment processing system within FunBill. It would allow the users to participate in one-to-one transactions.

- Dwolla's dev page: https://developers.dwolla.com/
- Has ACH and RTP options
- RTP is fast, funds available within seconds, irrevocable once processed
- ACH is bank-based, funds take 1-3 days, agreed upon processing / reversible
- ACH is commonly accepted, RTP is supported by limited parties
- $5000 weekly limit for personal, verified customers
- U.S. friendly
- No pre-built user interface for customers

Wise API

Formally known as TransferWise, Wise could be used to help process multiple currencies within the FunBill's built-in payment system. Although multiple currency usage is currently categorized as priority 2, it's best to keep options open if the chance arises.

- Wise's dev page: https://api-docs.transferwise.com/#wise-platform-api
- Connected with an application
- Have to create an account/profile for use
- Supports multiple currencies
- International friendly
- Pre-built user interface: mobile app for customers

JQuery API

JQuery will provide a lot of functionality and take away much of the bloat from the backend team. JQuery will provide an easy way to handle images from the users for receipt and profile picture storage, and we will also use it to implement sleek animations and transitions. JQuery is widely supported and used by several notable big companies as a lightweight framework on top of vanilla JavaScript.

- JQuery's documentation page: https://api.jquery.com/
- We will use JQuery's natural support for waiting to execute functions until specific parts of the DOM have loaded in order to provide a well-designed product.

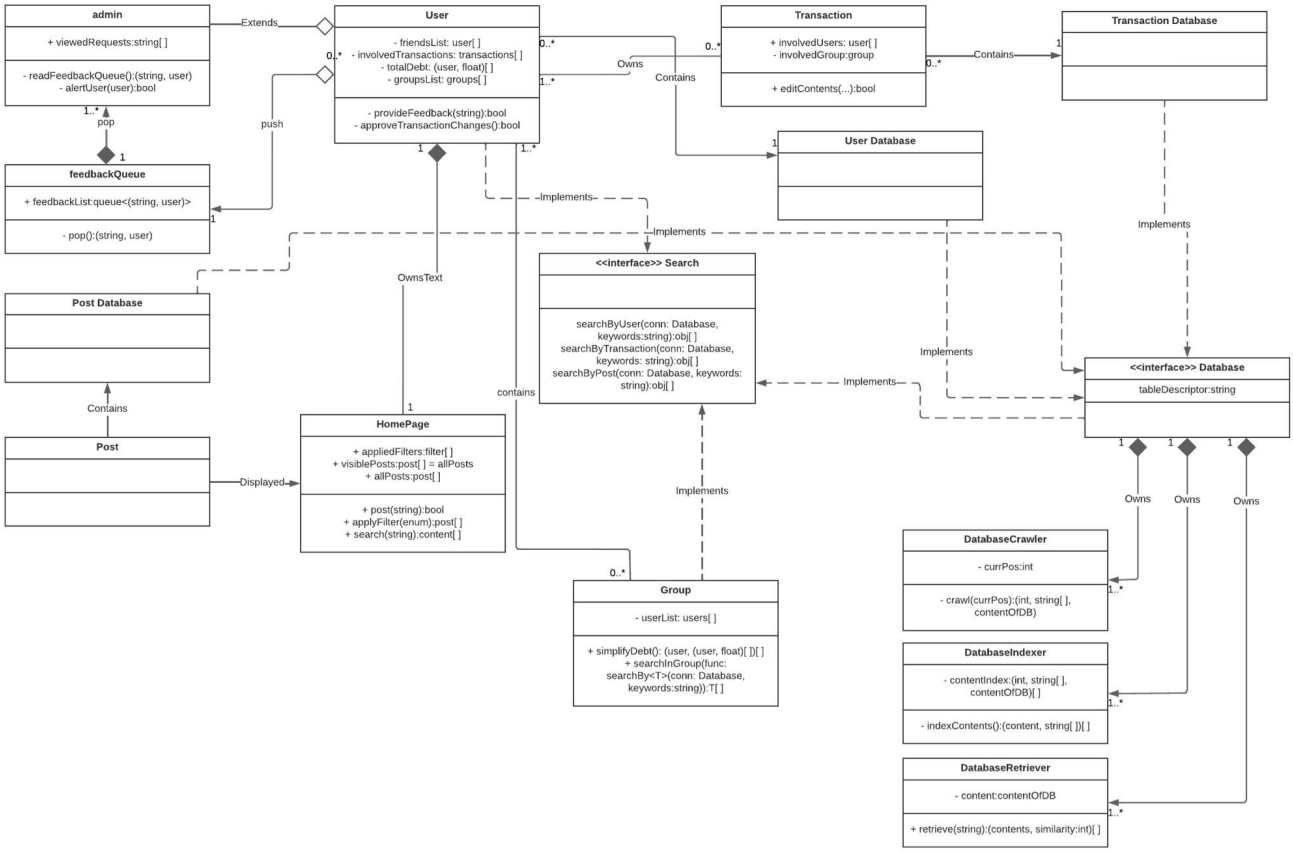
Back to Table of Contents

Section VI: High-Level UML Diagrams

UML Diagram

Link to the UML Diagram: [FunBill UML]

This UML displayed is focused on only the main high-level classes that pertain to the best Key Features the FunBill is intended to have implemented. The UML showcasing the entirety of FunBill is also located within the link given.



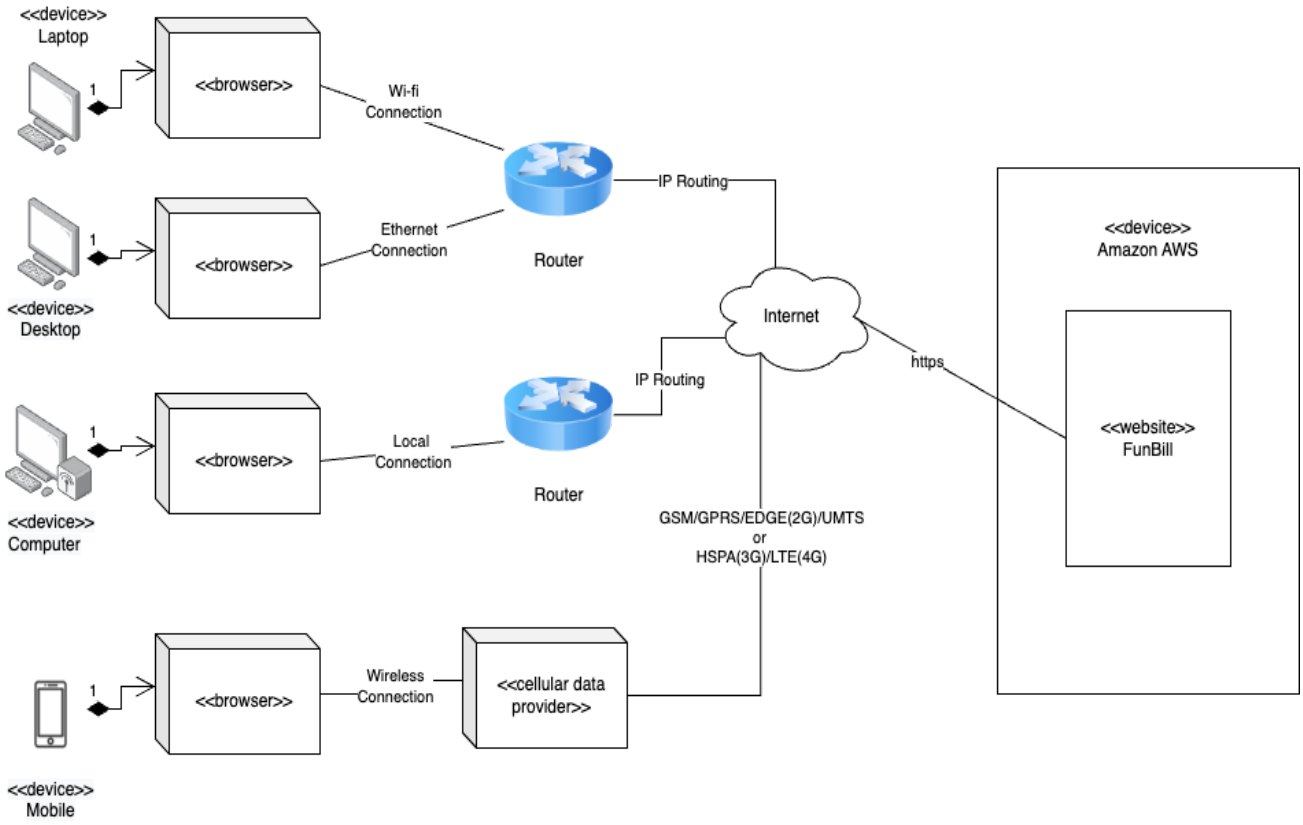
Back to Table of Contents

Section VII: High-Level Diagrams

Application Networks Diagram

Link to the diagram: FunBill AND

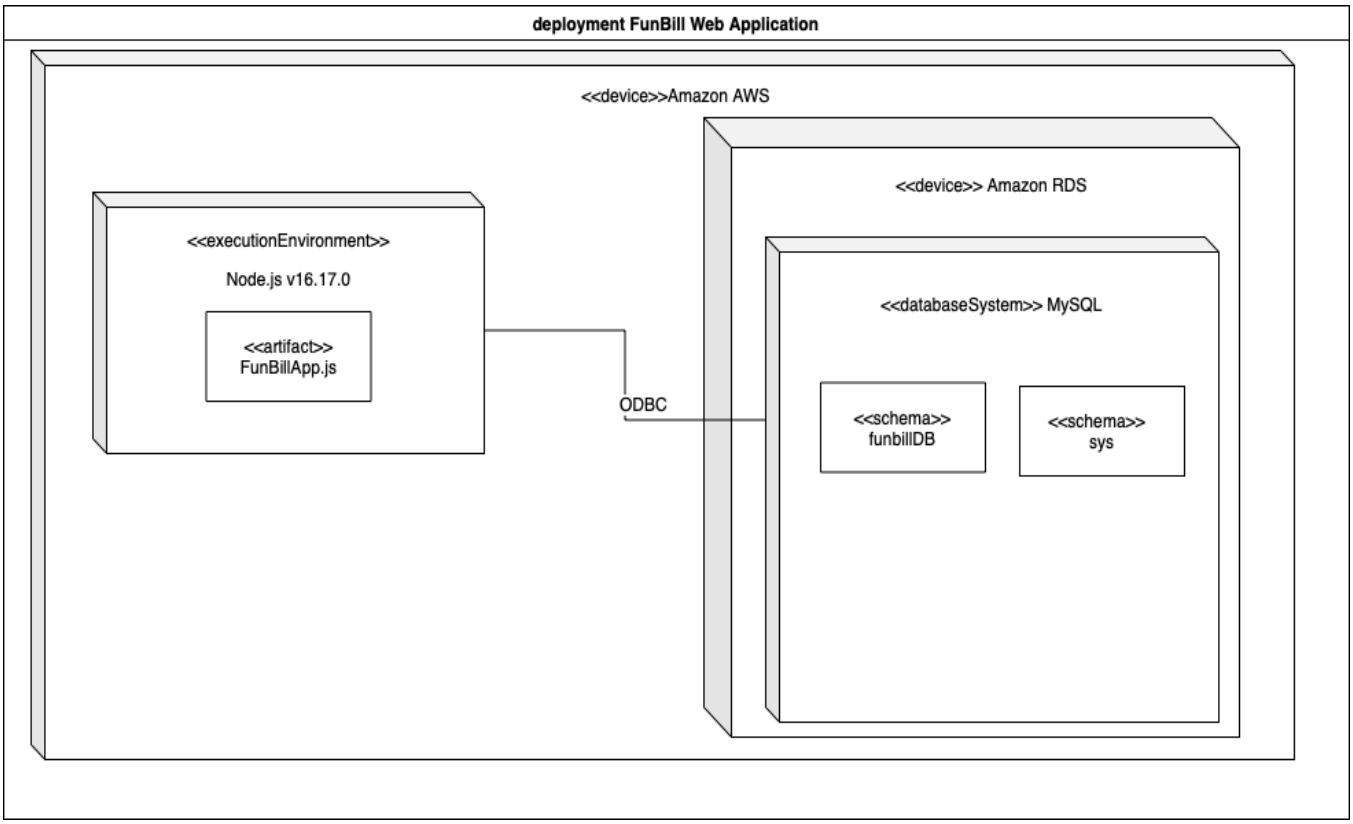
This diagram will describe the logical and physical networks used by our product from host to end systems. It shows protocols used for each service in your system, network configuration, security, gateways, firewalls, and proxies…



Back to Table of Contents

Deployment Diagram

Link to the diagram: FunBill DD
This diagram will show the architecture of the system a deployment (distribution) of software artifacts, or modules in our system, to deployment targets



Back to Table of Contents

Section VIII: Key Risks

1. **Skill and technical risks (APIs)**

   The team has the right set of skills to follow through with the overall product's design. With the use of APIs, the team can achieve certain functionalities without having to build them in-house from the ground up within a limited amount of development time. The challenge would be that the back-end team is new to designing programs with the utility of APIs developed in the professional world. Back-end team members plan to utilize APIs to achieve tasks such as payment systems and image processing. In order to utilize the APIs effectively, the team members will do research on the APIs and their documentation before using them. After research and comparison of multiple APIs that are similar to each other in terms of functionalities, one will be picked for each functionality. Through the use of APIs, the team is able to easily make use of certain functionalities while still lacking the skills to build them at good quality.

2. **Schedule and teamwork risks (Management)**

   Work is spread out amongst the team members based on category: front-end and back-end, with specific tasks and subcategories under the two. With a team of six, communication is necessary and it is essential to make sure that everyone is on the same page at all times throughout the development of the product. As to ensure that everyone has work that is to be done on schedule, we use Trello as a task management tool. With Trello, we are able to manage tasks and update their status in real-time in categories such as "to-do", "doing", "waiting on feedback", and "done". With help of Trello, the team is able to keep each other updated on the status of the tasks and ensure that the development of the program is on track appropriately.

3. **Teamwork risks (Google Drive and GitHub)**

   It is essential to have a central hub for the team to work on and share all the developed products and documentation. As to manage the progress and allow the team to work on a singular canvas of work, we use Google Drive for documentation and GitHub/Replit for developing code. Through the use of Google Docs, the team members are all able to hop on at any time and contribute to writing the documentation together. Before the documentation is made for the final use of submission, each team member can inspect the documentation before compiling it for submission. With the utilization of Github, the team is able to establish a central hub for the developing code of the project. Team members are able to share the working progress of the code, for both the back-end and front-end, on the GitHub repository.

Back to Table of Contents

4. **Legal/content risks**

   We do have the right set of tools to avoid any illegal use of content. The development team is taking precautions to avoid any strikes of copyrights by using only content and media that are free to use or originally designed by members of the team. For instance, the vector art that will be utilized on the front-end user interfaces will be originally drawn by members of the front-end team. Furthermore, it is to be considered an option to purchase content for use as to have rights to use any outside content, if need be.

Back to Table of Contents

Section IX: Project Management

Tasks were split and managed through the use of Discord and Trello. Discord has been the main source of communication amongst the group and Trello has been used to split up the work into smaller pieces, which members are then assigned to.

Tasks are first split by section, which is laid out in the milestone instructions, then split into subsections if there are multiple requirements. From there, members are assigned to a task, a due date is given, and a checklist is made. Most of the tasks were created by the team lead but can be/have been created by other group members. Once a task is complete, it is moved to “Waiting on Feedback” to await feedback. If others have looked over the work and is all good to go, the task is moved under the “Done” section. If a team member changes or comments on the milestone document, it is expected that the member makes a comment on Trello under its respective task.

There is a channel on the group’s Discord server, that shows a quick overview and agenda for the whole group. This way, group members can easily see what tasks they need to complete for the week. Group members can ask questions and update each other on the main channel.

Back to Table of Contents

Section X: Team Contribution

Team Member:	**Denyse Luzon**
Role:	Team Lead
Score:	8
Document Contribution:	Overall - Structured the document format/layout - Made comments/feedback for every section Functional Requirements - Transferred the data from M1 - Prioritized the functional requirements Project Management - Explained how tasks were distributed and managed - Organized a mix of Trello and Discord to communicate with the team
Prototype Contribution:	Code - Updated and tested server connections with the frontend's implementations - Reference links/buttons to their respective pages

Team Member:	**Bhagdeep Sandhu**
Role:	Database Master and Backend Team
Score:	10
Document Contribution:	Functional Requirements - Edited to fit the section requirements - Added new functional requirements Database Organization - Wrote down the database requirements - Created the ERD for the entirety of FunBill
Prototype Contribution:	Design - Created EER containing all the needed tables for this vertical prototype - Forward engineered the schema - Inserted sample data to be used in the search feature

Back to Table of Contents

Team Member:	**Faiyaz Chaudhury**
Role:	Backend Lead
Score:	10
Document Contribution:	Data Definitions - Transferred the data from M1 - Edited to fit the section requirements Database Organization - Documented findings when looking into implementing the Search/Filter feature APIs - Researched and documented the pros/cons to the JQuery API UML Diagram - Contributed to most of the diagram - Created a UML diagram for the entirety of FunBill - Created a narrowed down diagram to section fit the functionality we had that either our competitors did not have or we improved greatly upon Other Diagrams - Worked on Application Network Diagram - Worked on Deployment Diagram
Prototype Contribution:	Code - Created a script for the seach feature

Team Member:	**Kaung Nay Htet**
Role:	Backend Team
Score:	7
Document Contribution:	APIs - Researched and documented the pros/cons to the Dwolla API - Researched and documented the pros/cons to the Wise API UML Diagram - Contributed to some of the diagram - Helped to give feedback Other Diagrams - Worked on Application Network Diagram - Worked on Deployment Diagram Key Risks - Covered all 4 of the present risks
Prototype Contribution:	—

Back to Table of Contents

Team Member:	**Poornank Purohit**
Role:	Frontend Lead
Score:	10
Document Contribution:	Storyboards - 1, 2, 3, 7, 10, 13, 14, 15
Prototype Contribution:	Design - Created half of the design for the landing page Code - Created the layout for the landing page and home page

Team Member:	**Tolby Lam**
Role:	Frontend Team
Score:	10
Document Contribution:	Storyboards - 4, 5, 6, 8, 9, 11, 12
Prototype Contribution:	Design - Created half of the design for the landing page Code - Implemented the agreed-upon color scheme - Implemented chosen fonts

Back to Table of Contents