CSC 4370, Project 3

Crazy Coder's of the Codd

Sean Long - Team Lead
Glenn Murray
Brianna Hill
Osa Henry-Iyasere

Requirements

Project requirements

Group	Individual
Choose Leader	Describe how SCRUM benefitted the team
Choose Name	& helped solve problems
Produce Test Case	
Produce code snippets in PPT presentation	
Github repository everyone uploads to, with a history	
Video – 8 – 10 minutes in length	each person makes a contribution

All groups must complete Milestone #1, #2, and one of {#3,#4,#5}

Milestone #1

Homepage – describing the work, company, etc

User Registration – User registers with the company, including name, username, email, password
Information stored in DB

Milestone #2

Login page – User logs in, is verified and directed the Buyer's dashboard, Seller's dashboard, or Admin dashboard.

Milestone #3 (Choose 1)

Milestone #3:	Seller's dashboard
Milestone #4:	Buyer's dashboard
Milestone #5:	Admin dashboard

Software Requirements

Features

- Intuitive, easy to use navigation
- Forms incorporated
- Effective use of images, colors, type fonts, visual design
- It has to work!
- Cookies with sessions

Aspects

- Implementation of logic presented in presentation
- Application of good design principles

Pages

Home page

- Description of project
- What Company does
- Kind of service provided
- What's our company's competitive advantage
- What our business does to attract customers
- Has links to Registration and Login pages

User Registration Page

- User registration
- Automatic redirection to login screen once registered
- Password Encryption

FE Form Components

- first_name
- last_name
- email address
- phone
- password
- Password confirmation
- account holder type (buyer/seller/admin)

(? Leads to page collecting payment information

- Name
- Address
- Credit Card Number (Type can be inferred from number)
- Exp date
- Phone #
- Security Code

*Requires input validation

Login Page

Form fields

- username
- password

Seller Dashboard Page

Link to enter a property

Property cards (which are links to update information on cards)

Property Entry Page

Form fields

- Property owner id (Hidden)
- Name (required)
- Street Address (required)
- City (required)
- Zip (required)
- Build Date (required)
- Sq Footage (required)
- #Bedrooms (required)
- #Bathrooms (required)
- Selling Price(required)
- Picture (pic name) (optional)

DB design

Tables

Tables	*Fields*		*Field characteristics*
users			
	User_number		integer, internal, auto_increment
	Username		text, alphanumeric
	Password		text, encrypted
	Usertype	Buyer/Seller/Admin	Restricted text, internal
	Name_last		text, alpha
	Name_first		text, alpha
	Address		text, alphanumeric
	Phone_number		text, numeric

```
CREATE TABLE users
        ( user_number INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
         user_name VARCHAR(100) NOT NULL,
         first_name VARCHAR(255) NOT NULL,
         last_name VARCHAR(255) NOT NULL,
         password VARCHAR(255) NOT NULL,
         user_type VARCHAR(255) NOT NULL,
         email VARCHAR(255) NOT NULL,
         phone_number VARCHAR(255) NOT NULL,
         street_addr VARCHAR(255) NOT NULL,
         city_addr  VARCHAR(255) NOT NULL,
         state  VARCHAR(255) NOT NULL,
         zip VARCHAR(255) NOT NULL );
```

```
SHOW COLUMNS FROM users;
+----------------------+-----------------+-------+------+-----------+----------------------+
| Field                | Type            | Null  | Key  | Default   | Extra                |
+----------------------+-----------------+-------+------+-----------+----------------------+
| user_number          | int(11)         | NO    | PRI  | NULL      | auto_increment       |
| user_name            | varchar(100)    | NO    |      | NULL      |                      |
| first_name           | varchar(255)    | NO    |      | NULL      |                      |
| last_name            | varchar(255)    | NO    |      | NULL      |                      |
| password             | varchar(255)    | NO    |      | NULL      |                      |
| user_type            | varchar(255)    | NO    |      | NULL      |                      |
| email                | varchar(255)    | NO    |      | NULL      |                      |
| phone_number         | varchar(255)    | YES   |      | NULL      |                      |
| street_addr          | varchar(255)    | YES   |      | NULL      |                      |
| city_addr            | varchar(255)    | YES   |      | NULL      |                      |
| state                | varchar(255)    | YES   |      | NULL      |                      |
| zip                  | varchar(255)    | YES   |      | NULL      |                      |
+----------------------+-----------------+-------+------+-----------+----------------------+
```

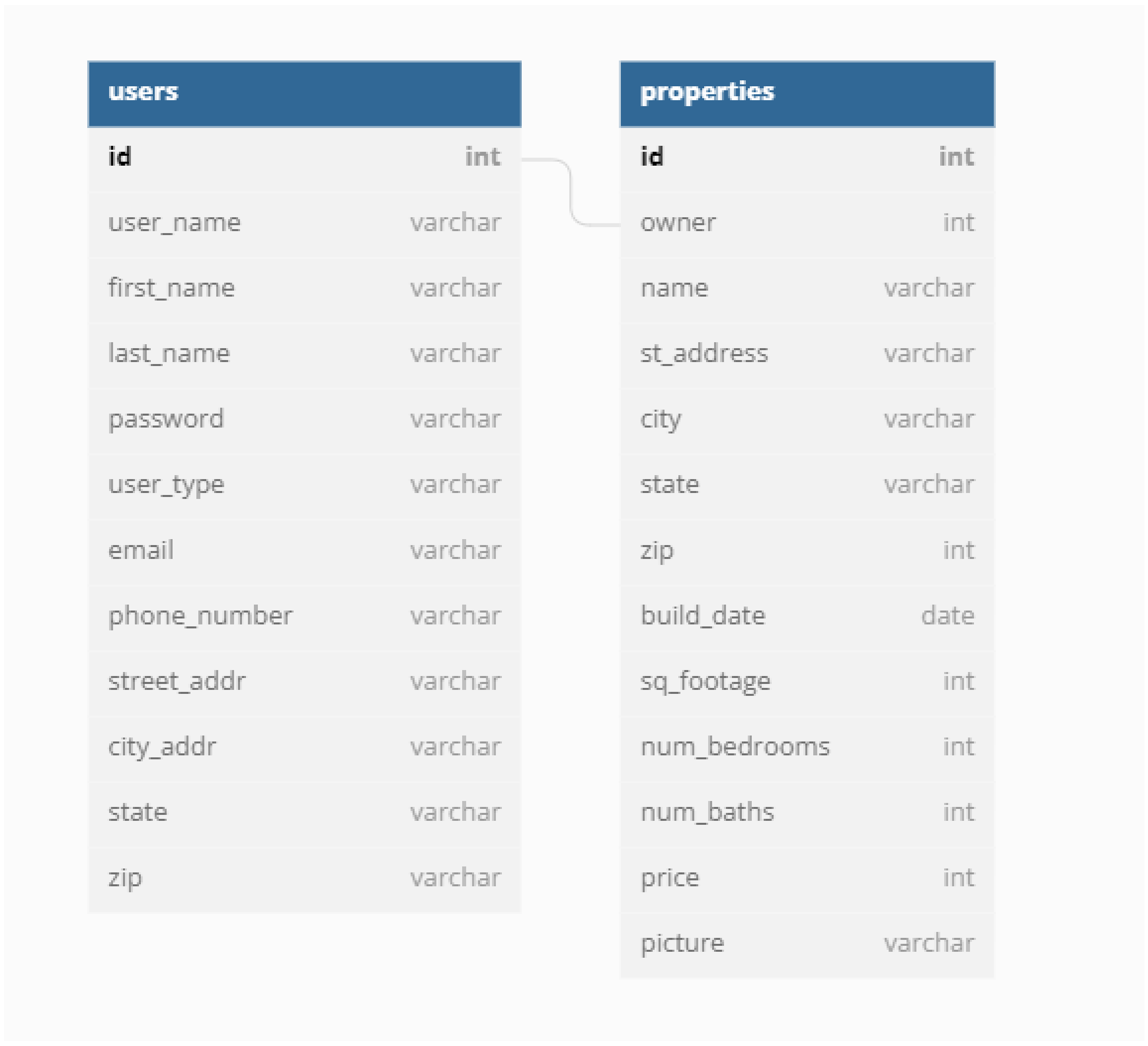
properties

id	integer, internal, auto_increment, required
owner	integer, foreignKey, required
name	varchar(255), alphanumeric, required
st_address	varchar(255), alphanumeric, required
city	varchar(100), alpha, required
zip	int, numeric, required
build_date	date, required
sq_footage	integer, required
num_bedrooms	integer, required
num_baths	float, required
selling_price	integer, required
picture	varchar(100), optional

```
CREATE TABLE properties
    ( id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
      owner INT NOT NULL REFERENCES users(user_number),
      name VARCHAR(100) NOT NULL,
      st_address VARCHAR(255) NOT NULL,
      city VARCHAR(100) NOT NULL,
      state VARCHAR(100) NOT NULL,
      zip INT NOT NULL,
      build_date DATE NOT NULL,
      sq_footage INT NOT NULL,
      num_bedrooms INT NOT NULL,
      num_baths DEC(3,1) NOT NULL,
      price INT NOT NULL,
      picture VARCHAR(100) );
```

```
SHOW COLUMNS FROM properties;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| id           | int(11)      | NO   | PRI | NULL    | auto_increment |
| owner        | int(11)      | NO   |     | NULL    |                |
| name         | varchar(100) | NO   |     | NULL    |                |
| st_address   | varchar(255) | NO   |     | NULL    |                |
| city         | varchar(100) | NO   |     | NULL    |                |
| state        | varchar(100) | NO   |     | NULL    |                |
| zip          | int(11)      | NO   |     | NULL    |                |
| build_date   | date         | NO   |     | NULL    |                |
| sq_footage   | int(11)      | NO   |     | NULL    |                |
| num_bedrooms | int(11)      | NO   |     | NULL    |                |
| num_baths    | decimal(3,1) | NO   |     | NULL    |                |
| price        | int(11)      | NO   |     | NULL    |                |
| picture      | varchar(100) | YES  |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
```

Schema relationships



users	
id	**int**
user_name	varchar
first_name	varchar
last_name	varchar
password	varchar
user_type	varchar
email	varchar
phone_number	varchar
street_addr	varchar
city_addr	varchar
state	varchar
zip	varchar

properties	
id	**int**
owner	int
name	varchar
st_address	varchar
city	varchar
state	varchar
zip	int
build_date	date
sq_footage	int
num_bedrooms	int
num_baths	int
price	int
picture	varchar

Testing

Test Cases

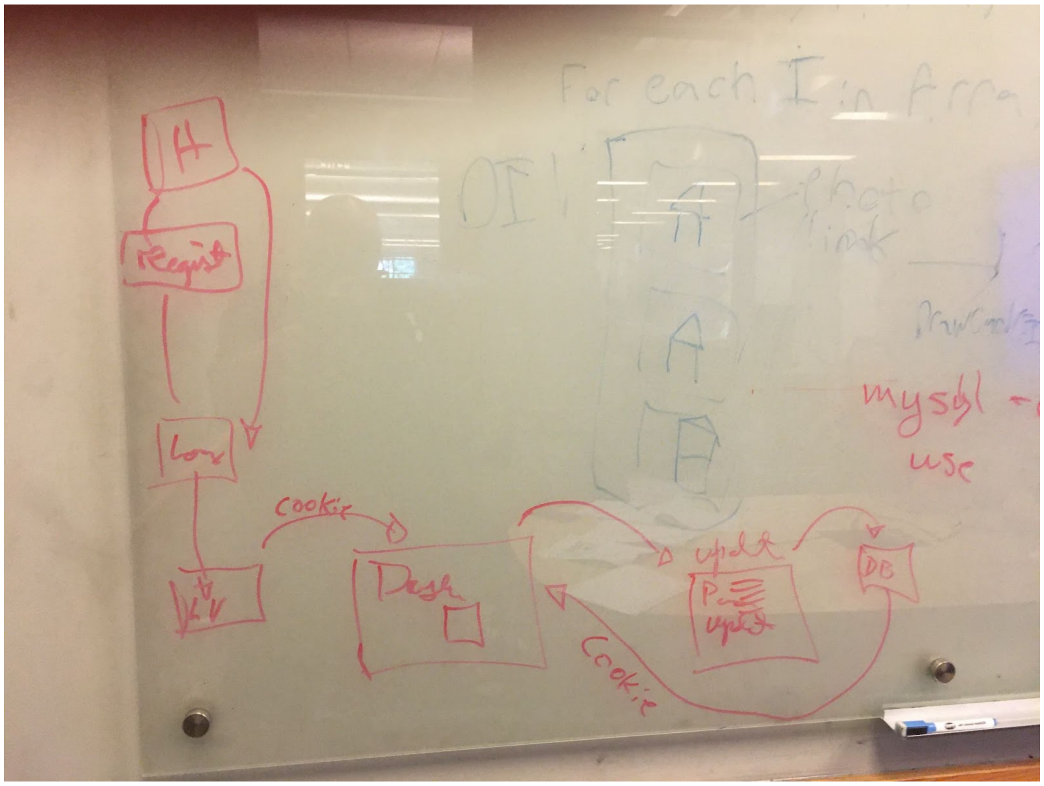
1. Check to ensure all input conforms to input criteria, and non-conforming is rejected (Login, Registration, Payment page)
2. Ensure a new user is created in DB when entered on the registration page.
3. (optional) Check for SQL injection attack [see 1]
4. Ensure new users are taken to the correct dashboard, on login or registration.
5. Rejected Cases: Load/performance tests not feasible
6. Ensure dashboard conforms to the specific user on a successful login/registration
7.

Milestone 3 Frontend-> Backend
Step 1 returnProperties(userID) Returns array properties for that user This is backend
Step 2 draw photos(Array A) Populate div with house pics Onclick() loadCard(array[i])
Step 3 LoadCard(Array[i]) Displays all table info formatted into card Has returnToPhotos() button
--Milestone 4 Frontend ->

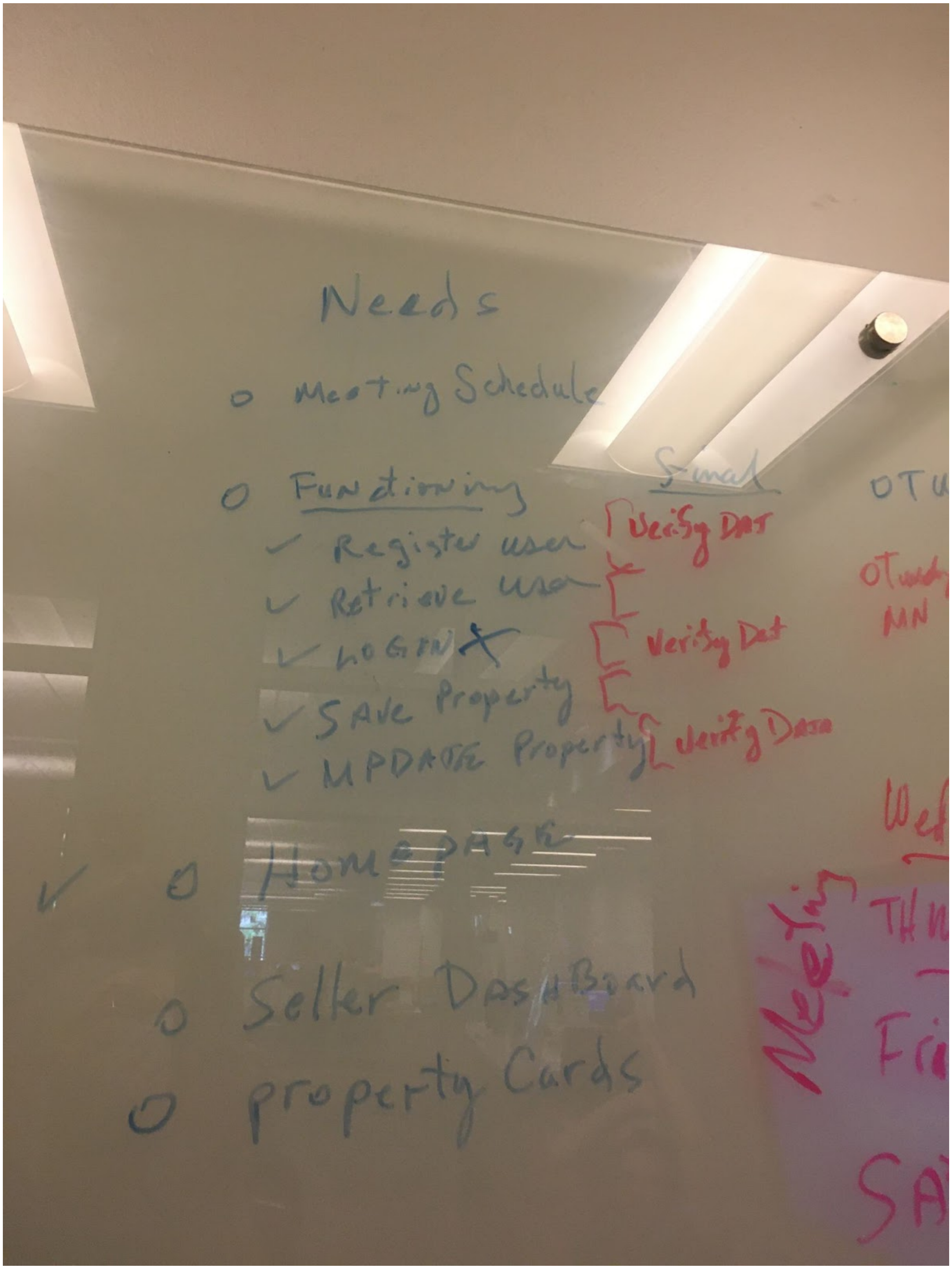
```
 Backend EditCard()
     updates innerHTML
     SubmitChanges()
     updates DB and returnsToPhotos()
```

Design Notes

Page flow



Needs list - pages



Needs

- o Meeting Schedule
- o Functioning
  - ✓ Register user — Verify Dat
  - ✓ Retrieve user
  - ✓ LOGIN ✗ — Verify Dat
  - ✓ SAVE Property
  - ✓ UPDATE Property — Verify Data
- ✓ o HomePAGE
- o Seller DashBoard
- o property Cards

Final

OTU

OTuesday MN

Wed

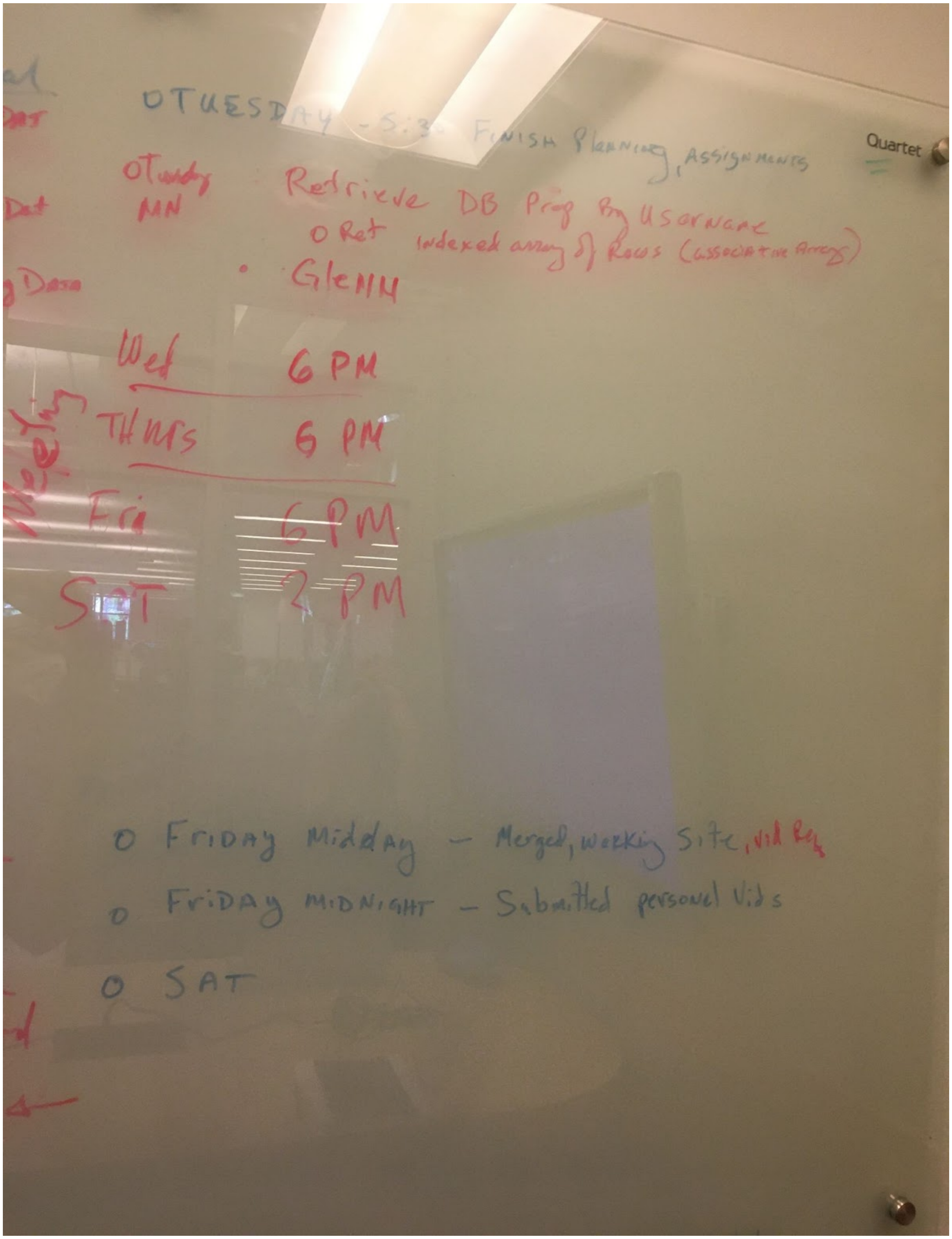
Meeting

TH

Fri

SA

'Standup' Meeting Schedule



o TUESDAY - 5:30 FINISH Planning, ASSIGNMENTS

o Tuesday MN Retrieve DB Prop By Username
  o Ret indexed array of Rows (associative Arrays)
  • Glenn

Wed 6 PM

Thurs 6 PM

Fri 6 PM

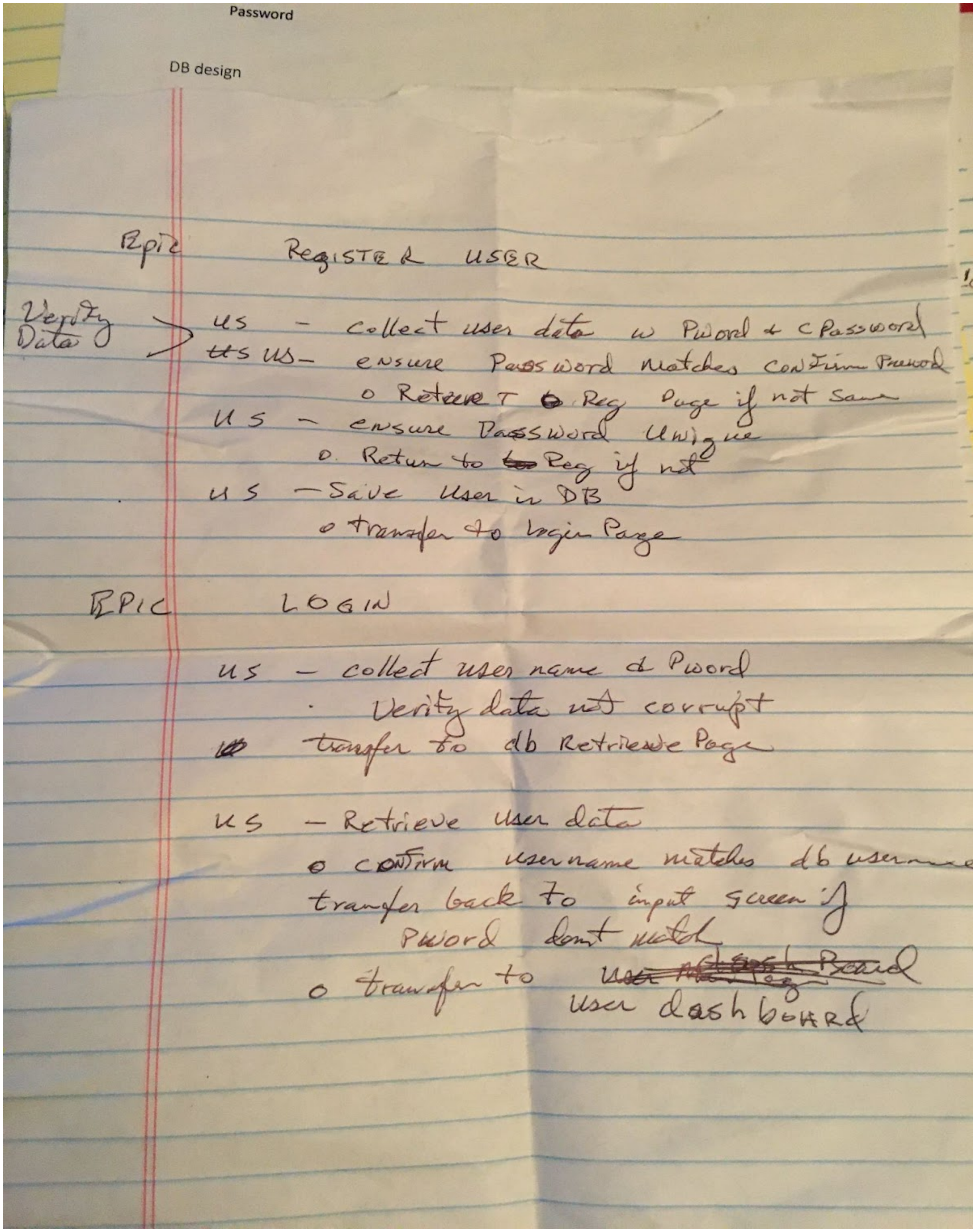
SAT 2 PM

o Friday Midday – Merged, working Site, vid Req

o Friday MIDNIGHT – Submitted personal Vids

o SAT

Epics and User Stories



Password

DB design

Epic REGISTER USER

Verify Data

us - collect user data w Pword & cPassword

~~us~~ us - ensure Password matches confirm Pword

o Return to Reg Page if not Same

us - ensure Password unique

o Return to ~~to~~ Reg if not

us - Save User in DB

o transfer to login Page

EPIC LOGIN

us - collect user name & Pword

. Verify data not corrupt

~~us~~ transfer to db Retrieve Page

us - Retrieve user data

o confirm username matches db username

transfer back to input screen if Pword dont match

o transfer to ~~User Master Board~~ user dashboard

Tables

EPic   Enter Property

us · ~~Enter~~ Colled ~~[illegible]~~ Property data   form
> Verify data
st   o ~~and~~ transfer to Property Entry Page

us   Enter Prop data into db
o Return to users dash board

~~[illegible]~~

Epic   Edit Property

us   collect ~~[illegible]~~ Prop Id
> Verify

us   ~~[illegible]~~ Retrieve Prop data From db
o if not found Return to dash board

US   Fill in Form From db data

~~Post~~   ~~transfer~~
US   user Edits data
xFer to store Page

post

Us   Store data c̄ db
X Fer to dash Board