SW Engineering CSC648/848 Section 01 Spring 2017
Team 05

Gator 2 Gator

Milestone 2

March 15, 2017

Andrew Lesondak
Tim Bauer
David Rodriguez
James Lee
Mayara Dusheyko
Chengjiu Hong
Jeffrey Hu

History

Initial Draft	v1.0

1. Use Cases

2.1 – Browse/Search Listing (Guest, Registered User - Buyer)

Alice is a sophomore student at SFSU. She is double majoring in Psychology and Computer Science. After she enrolls for all her seven classes, she realizes that the list of required books is very long and that each of them costs a small fortune. Likely, as she complains about the amount of money she will need to spend in books this semester, her roommate who is also a SFSU student, refers her to the SFSU Gator 2 Gator buy and sell application in which she could easily look for the books she needs, buy them for a fraction of the cost and receive them much faster than if she were to buy them from another online source. In addition to that, Gator 2 Gator provides the safety of having buyers and sellers meeting on a public location on SFSU Campus. Alice feels very hopeful about this, especially about the security that comes from trading on SFSU Campus, since she has previously had problems with in person transactions with unknown sellers. She navigates to the web application and can browse through the items. She finds all the books she is looking for and in addition to that, she finds great deals on furniture for her new apartment. When she decides which items to buy, since she is a **Guest User**, she is prompted to register so she can contact the seller, find out about the meeting location and finish the transaction.

2.2 - Post/Edit/Delete Listing (Registered User - Seller)

Ron is moving back to his hometown in Germany after spending a year as an exchange student at SFSU. He has various pieces of furniture such as a bed, an office table and chair, as well as books from his last semester that he no longer needs. Since Ron is moving back to Germany and cannot take his furniture with him, he decides to sell all of his items using the web application for buying and selling for SFSU students that one of his friends introduced him to when he started studying at SFSU. Ron finds this application very useful because he can be confident that he will only make transactions with students and will avoid getting into problems when he sells the items. Therefore, as a **Registered User** Ron can easily create a listing for the items he would like to sell. After that, Ron can view all of his posted items and choose to edit or delete any of them.

2.3 - Delete User/Delete Listing (Admin)

Jacky is the system **Admin** in the SFSU buy and sell application. She is responsible for preventing bad users from disrupting the application. As an Admin, she can view all the users, their listings and can delete any of them. If a Registered User posts improper content, Jacky promptly removes the listing from the application in order to ensure that all the Guests and Registered Users have access to only the listings that are suitable for the application. Further, Jacky may remove any Registered Users from the system so that they are no longer able to log into the application in the future. When the Gator 2 Gator application employs a new system Admin, Jacky is able to grant Admin privileges to them, so that they can start managing the system.

2. Data Definitions

1) Guest User
   a) Shall be able to register an account using SFSU e-mail
   b) Shall be able to filter items
   c) Shall be able to browse items

2) Registered User
   a) Shall be able to provide a username or e-mail address, and a password to sign in
   b) Shall be able to browse items
   c) Shall be able to buy items
   d) Shall be able to filter items
   e) Shall be able to sell items
   f) Shall be able to send and receive message from Registered User

3) Admin
   a) Shall be able to provide a username or e-mail address, and a password to sign in
   b) Shall be able to browse items
   c) Shall be able to remove items
   d) Shall be able to delete Registered User
   e) Shall be able to send and receive message from Registered User
   f) Shall be able to filtered items
   g) Shall be able to grant permission to other Admin

4) ItemPost
   a) Price: the amount of money required for the item
   b) Description: a written representation of the item
   c) Post date: the date which the Registered User-seller make the item for sell
   d) Item name: an identification by which the item is known
   e) Category: a division of items regarded as having particular shared characteristics.
   f) Photo: photo that are relevant to the item

5) Message
   a) Subject: a heading or a brief summary of the matters between Registered Users
   b) Sender: a Registered User who sends the message

6) MessageMetadata
   a) Message: a body of paragraph describing the matters between two Registered Users
   b) Participant: a Registered User who receives the message

3. Functional Requirements

Priority 1:

Guest Users

- Shall be able to browse the site and view item listings.
- Shall be able to create an account an account or register as a user.
- Shall be required to provide a valid SFSU email when registering.

Registered Users - Buyers

- Shall be able to browse the website and filter through posts
- Shall have messaging option to contact the seller.

Registered Users - Sellers

- Shall be able to post an item to sell.
- Shall have a messaging option to reply to buyers.
- Shall be able to browse the website and filter through postings.
- Shall be able to edit/delete their post.

Admin

- Shall be able to browse and filter through posts.
- Shall be able to delete posts.
- Shall be able to delete user accounts.
- Shall be able to promote new Admin.

Priority 2:

Guest Users

- Shall be able flag prohibited posts.

Registered Users - Buyers

- Shall be able to block users.
- Shall be able to rate sellers.

Registered Users - Sellers

- Shall be able to block users.
- Shall be able to rate buyers.

Admin

- Shall be able to message registered users.

Priority 3:

Guest Users

- Shall be able to add posts to a list of ‘favorites’.

Registered Users - Buyers

- Shall be able to add posts to a list of ‘favorites’.

Registered Users - Sellers

- Shall be able to have postings seen only by certain users.

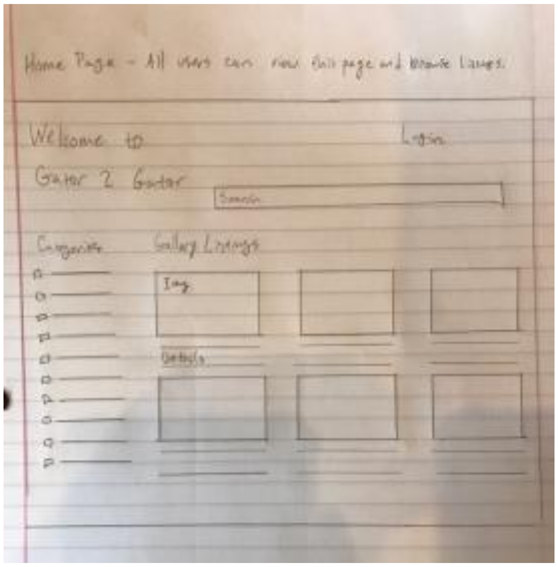
4. List of Non-Functional Specs

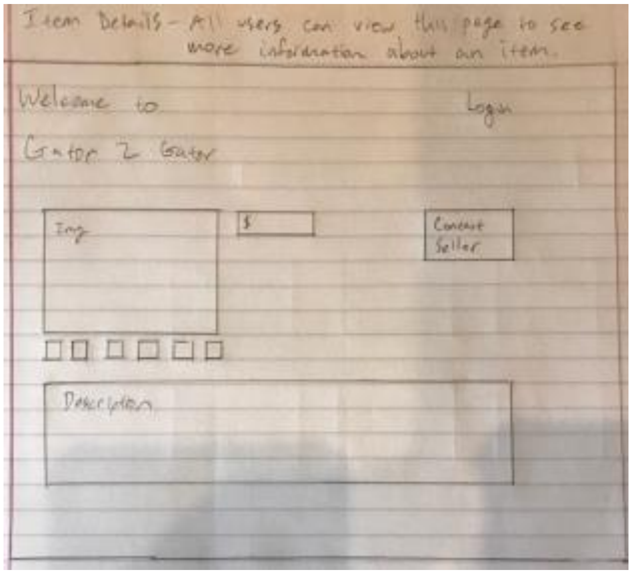
1. Application shall be developed using class provided LAMP stack
2. Application shall be developed using pre-approved set of SW development and collaborative tools provided in the class. Any other tools or frameworks must be explicitly approved by Anthony Souza on a case by case basis.
3. Application shall be hosted and deployed on Amazon Web Services as specified in the class
4. Application shall be optimized for standard desktop/laptop browsers, and must render correctly on the two latest versions of all major browsers: Mozilla, Safari, Chrome.
5. Application shall have responsive UI code so it can be adequately rendered on mobile devices but no mobile native app is to be developed
6. Data shall be stored in the MySQL database on the class server in the team's account
7. Application shall be served from the team's account
8. No more than 50 concurrent users shall be accessing the application at any time
9. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.
10. The language used shall be English.
11. Application shall be very easy to use and intuitive. No prior training shall be required to use the website.
12. Google analytics shall be added
13. Messaging between users shall be done only by class approved methods to avoid issues of security with e-mail services.
14. Pay functionality (how to pay for goods and services) shall not be implemented.
15. Site security: basic best practices shall be applied (as covered in the class)
16. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development
17. The website shall prominently display the following text on all pages *"SFSU Software Engineering Project, Spring 2017. For Demonstration Only"*. (Important so as to not confuse this with a real application).
18. Payment exchanges will not be done by the site and all exchanges shall be done by the user's choosing.
19. Browser versions that will be supported are Google Chrome, Mozilla Firefox, and Safari

5. UI Mockups and Storyboards

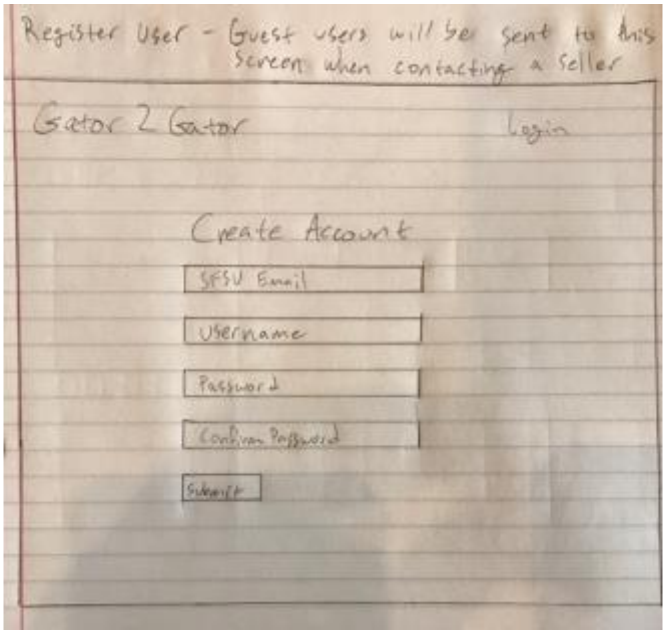
(**Browse/Search Listing**) Alice goes to Gator 2 Gator for the first time, she browses the items available for sale, filters by books, selects one, and attempts to contact the seller to make the purchase.

1) Arrives at homepage and makes a search/filter to find specific items
2) Clicks to see more information on the listed item.



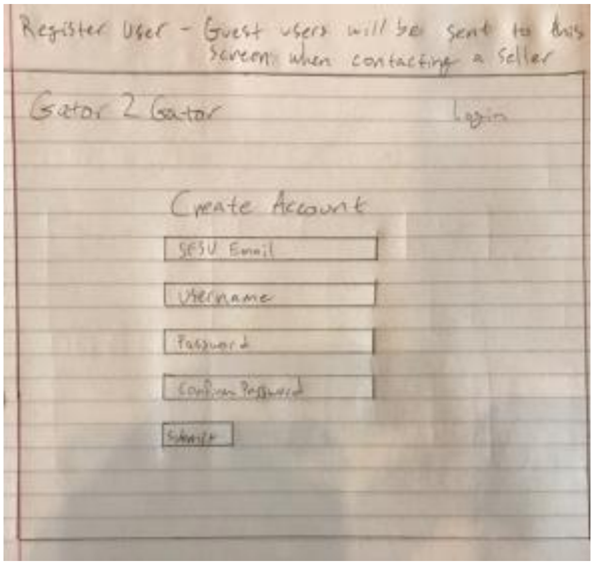

Item Details - All users can view this page to see more information about an item.
Welcome to
Gator 2 Gator
Login
Img
$
Contact Seller
Description

3) She finds the items she wants to purchase, so she clicks contact seller and is prompted to register.

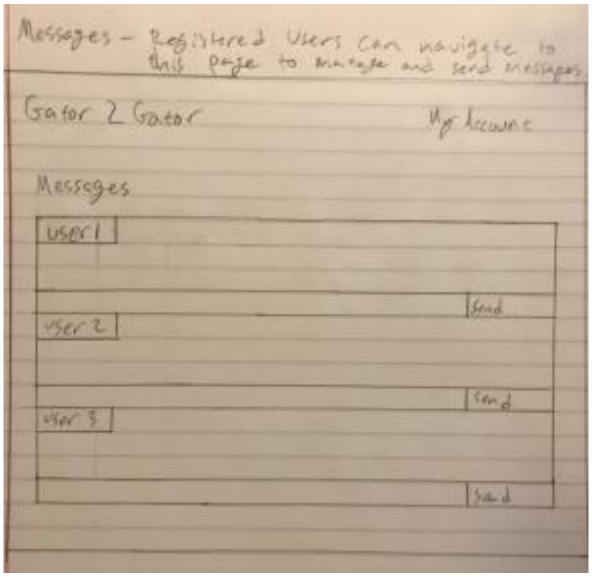

Register User - Guest users will be sent to this screen when contacting a seller
Gator 2 Gator
Login
Create Account
SFSU Email
Username
Password
Confirm Password
Submit

(**Registration-Messaging**) Mary has found a great piece of furniture that she would like to buy, however as she attempts to contact the seller she is prompted to register, and after that she can successfully message the seller about the item.

1) Mary clicks to contact the seller and is prompted to register as a user.
2) She can now send a message to the seller to initiate the purchase.



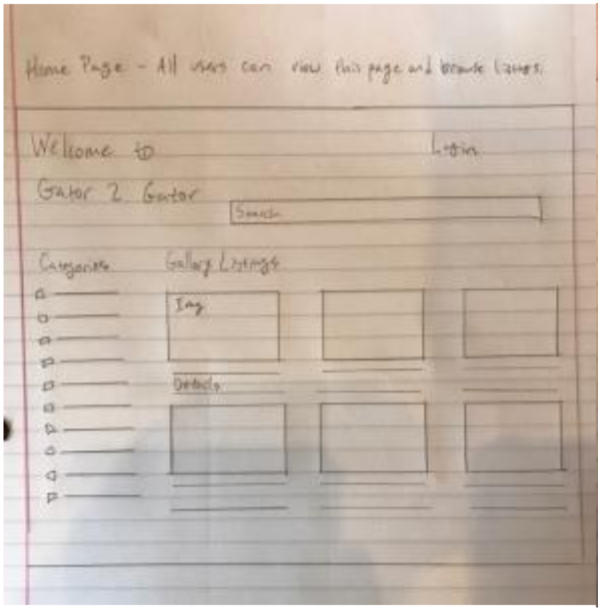
Register User - Guest users will be sent to this screen when contacting a seller

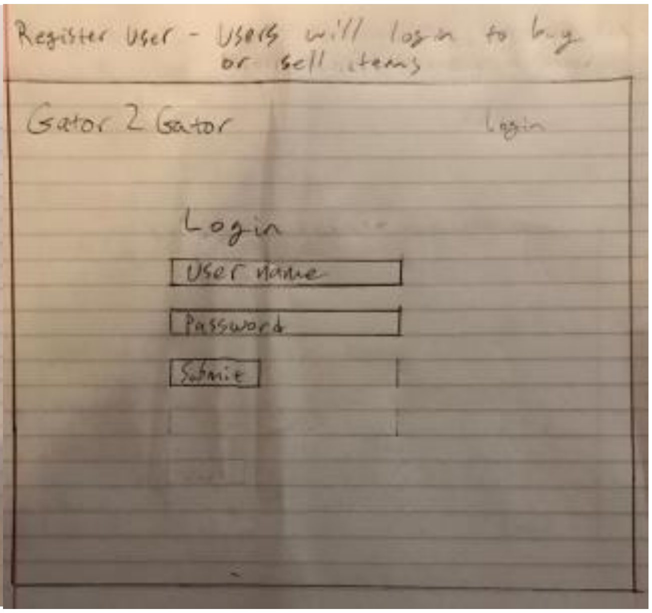


Messages - Registered Users can navigate to this page to manage and send messages.

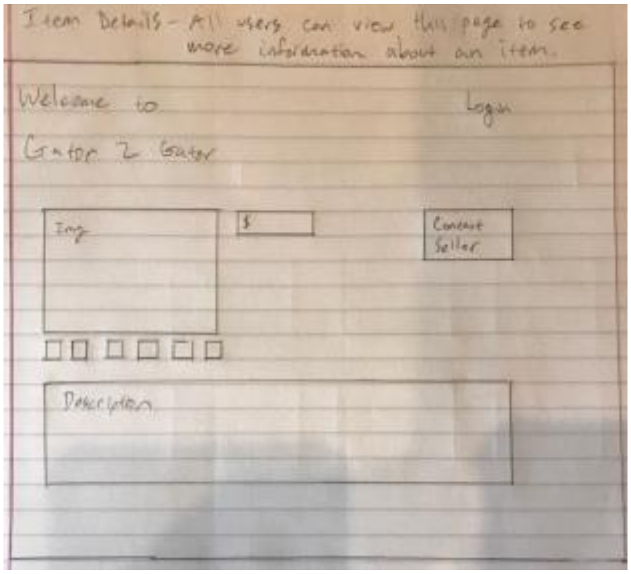
(Post Listing) Ron is a Registered User of Gator 2 Gator that needs to sell some furniture he no longer needs. He goes to the main page, logs in, and posts some new listings.

1) He clicks to login from the Home Page.
2) Provides login credentials.



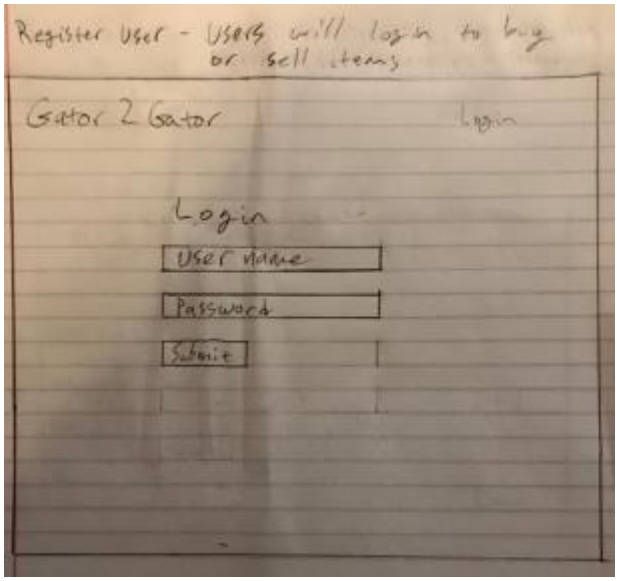


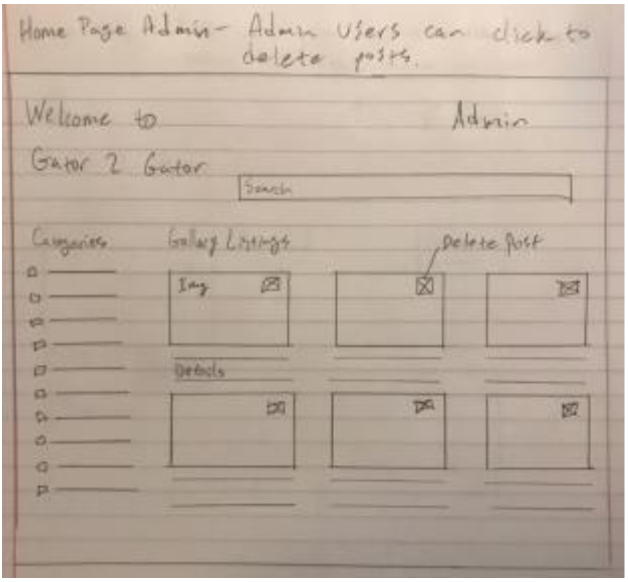
3) Clicks to post an item.


Item Details - All users can view this page to see more information about an item.
Welcome to
Gator 2 Gator
Login
Img
$
Contact Seller
Description

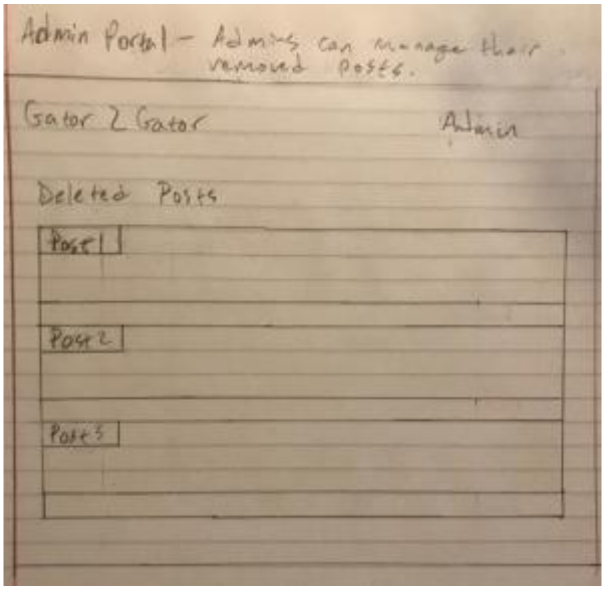
(**Delete Listing Admin**) - Jacky is inspecting the posted listings and sees several posts with content that is not allowed on Gator 2 Gator. She deletes the listings and warns the Registered Users who posted them.

1) Jacky accesses her account by logging in
2) She finds the post which she would like to remove.





3) Jacky can manage the deleted posts from her Admin account.



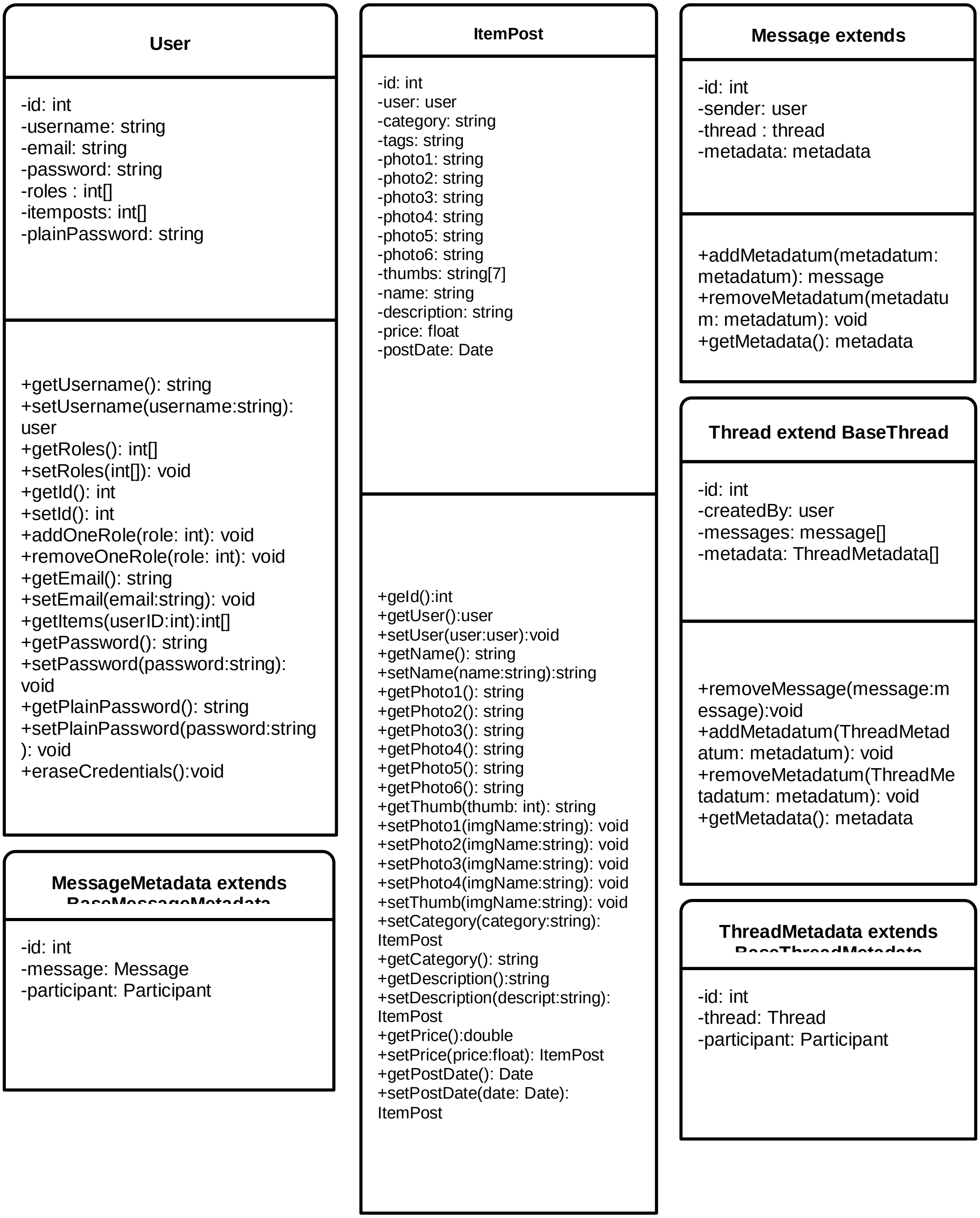
6. High Level Architecture

- Linux – The operating system the site will be hosted and deployed from is Ubuntu Linux Server version 16.0.4 provided by Amazon Web Services.
- Apache – The web server application installed and utilized is Apache (Current Version).
- MySQL – For database management this project will implement MySQL Version 5.7.
- PHP – The primary scripting language for this deployment is PHP version 7.0.13.
- Symfony – The full-stack framework this project will be implementing is Symfony version 3.2.
- MySQL Workbench - The primary tool for accessing and interfacing server side MySQL is MySQL workbench version 6.3.
- SSH – Accessing the remote server will be accomplished using OpenSSH version 7.2.
- PhpStorm – The team's primary Integrated Development Environment is PhpStorm Version 2016.3 by Jet Brains under student license.
- Git – Version control will be achieved using Git version 2.7.4
- GitHub – Online Version Control and management will be provided through github.com.
- Google Analytics – Website data collection and analytics services will be provided by Google's Analytics Reporting API version 4.
- Search will be conducted using %like function implemented by DBMS.
- Database schemas are, saleItem, regUser, p2pMsg, imgData and counters for unique IDs.
- Sorting and Filtering will be conducted using MySQL queries on data members.

7. High Level UML Diagrams

itemPost	User	counters	message	imgData
itemID: int(key)	userId: int(key)	usercount: int	msgId: int(key)	imgID: int(key)
photo1: string	username: string	postcount: int	userId: int(foreign key)	imgName: string
photo2: string	email: string	msgcount :int	destuser: int	imgHeight: int
photo3: string	password: string(encrypted)	imgcount: int	timestamp: int	imgWidth: int
photo4: string	accessflag: int		deliveredflag: int	imgThumb: string
photo5: string			msgtext: string	imgFormat: string
photo6: string			msgTitle: string	imgTags: string
mainthumb: string				
thumb1:string				
thumb2: string				
thumb3: string				
thumb4: string				
thumb5: string				
thumb6: string				
descript: string				
cat: string				
tags: string				
userId:int(foreign key)				
itemDate: Date				
itemPrice: double				
itemTitle: string				

7b. UML Diagrams



User

-id: int
-username: string
-email: string
-password: string
-roles : int[]
-itemposts: int[]
-plainPassword: string

+getUsername(): string
+setUsername(username:string): user
+getRoles(): int[]
+setRoles(int[]): void
+getId(): int
+setId(): int
+addOneRole(role: int): void
+removeOneRole(role: int): void
+getEmail(): string
+setEmail(email:string): void
+getItems(userID:int):int[]
+getPassword(): string
+setPassword(password:string): void
+getPlainPassword(): string
+setPlainPassword(password:string): void
+eraseCredentials():void

MessageMetadata extends BaseMessageMetadata

-id: int
-message: Message
-participant: Participant

ItemPost

-id: int
-user: user
-category: string
-tags: string
-photo1: string
-photo2: string
-photo3: string
-photo4: string
-photo5: string
-photo6: string
-thumbs: string[7]
-name: string
-description: string
-price: float
-postDate: Date

+geId():int
+getUser():user
+setUser(user:user):void
+getName(): string
+setName(name:string):string
+getPhoto1(): string
+getPhoto2(): string
+getPhoto3(): string
+getPhoto4(): string
+getPhoto5(): string
+getPhoto6(): string
+getThumb(thumb: int): string
+setPhoto1(imgName:string): void
+setPhoto2(imgName:string): void
+setPhoto3(imgName:string): void
+setPhoto4(imgName:string): void
+setThumb(imgName:string): void
+setCategory(category:string): ItemPost
+getCategory(): string
+getDescription():string
+setDescription(descript:string): ItemPost
+getPrice():double
+setPrice(price:float): ItemPost
+getPostDate(): Date
+setPostDate(date: Date): ItemPost

Message extends

-id: int
-sender: user
-thread : thread
-metadata: metadata

+addMetadatum(metadatum: metadatum): message
+removeMetadatum(metadatum: metadatum): void
+getMetadata(): metadata

Thread extend BaseThread

-id: int
-createdBy: user
-messages: message[]
-metadata: ThreadMetadata[]

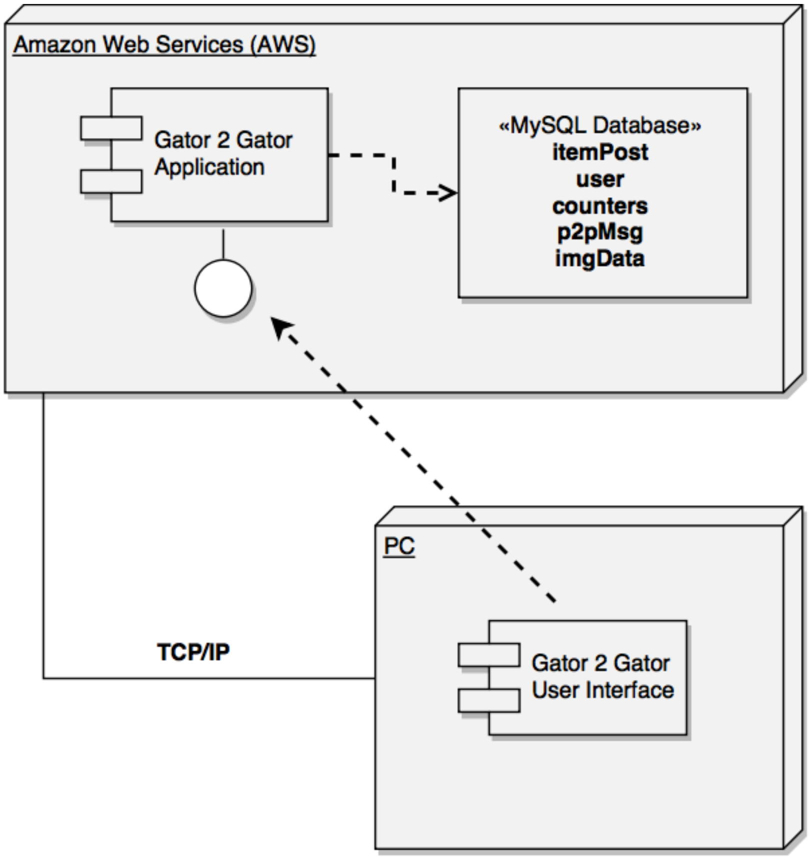
+removeMessage(message:message):void
+addMetadatum(ThreadMetadatum: metadatum): void
+removeMetadatum(ThreadMetadatum: metadatum): void
+getMetadata(): metadata

ThreadMetadata extends BaseThreadMetadata

-id: int
-thread: Thread
-participant: Participant

Note: BaseMessageMetadata, BaseThreadMetadata, BaseThread and BaseMessage are part of Symfony framework.

UML Component and Deployment Diagram


Amazon Web Services (AWS)
Gator 2 Gator Application
«MySQL Database»
itemPost
user
counters
p2pMsg
imgData
TCP/IP
PC
Gator 2 Gator User Interface

8. Key Risks

1. *skills* risks
   - The team is divided between a lot of experience with PHP and no experience with PHP. This presents the challenge of bringing everyone together to find a middle ground of knowledge. Tim, our CTO, has been very patient and has been providing the rest of the team with personal instructions and whiteboard micro lectures to get everyone up to speed.

2. *schedule* risks
   - This is our team's favorite challenge. We all have our own lives outside of work and therefore communication is extremely sensitive. We meet in our Slack chat group almost every day to keep up with each other. Our physical meeting dates are typically Wednesday afternoons.

3. *technical* risks
   - No technical risks.

4. *teamwork* risks
   - Keeping everyone organized and on point is the challenge. Due to our busy schedules, we are not able to come together for team building. This has to be done through our other means of communication. We all get along and are very flexible with each other. When one of us cannot make it to a meeting, someone else is making sure to have notes for them to follow along with at a later time.

5. *legal/content* risks
   - No legal risks. The team has proper licensing to all the development tools it is using.

9. Team Organization

Andrew Lesondak, CEO

Tim Bauer, CTO

David Rodriguez, Back-end Developer

James Lee, Front-end Developer

Mayara Dusheyko, Back-end Developer

Chengjiu Hong, Back-end Developer

Jeffrey Hu, Front-end Developer