Ctrl.Alt.Gift

Team East Atlanta Santa

Stephanie, Winston, Will, Zicheng
st234pa, wvenderbush, wostlund, zzcnick
Github url: https://github.com/wostlund/softdevFinalProj

Abstract: We will create a Secret Santa gift giving platform that is better than Elfster. This tool will facilitate Secret Santa gift exchanges and make it really fun and cool.

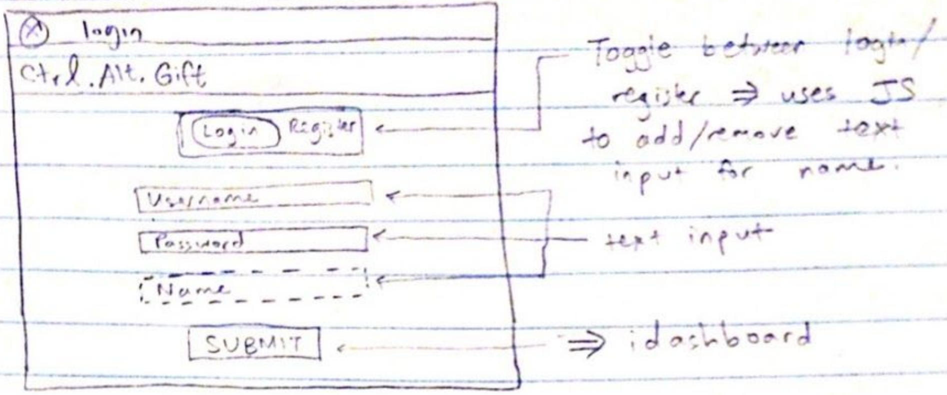
Features:

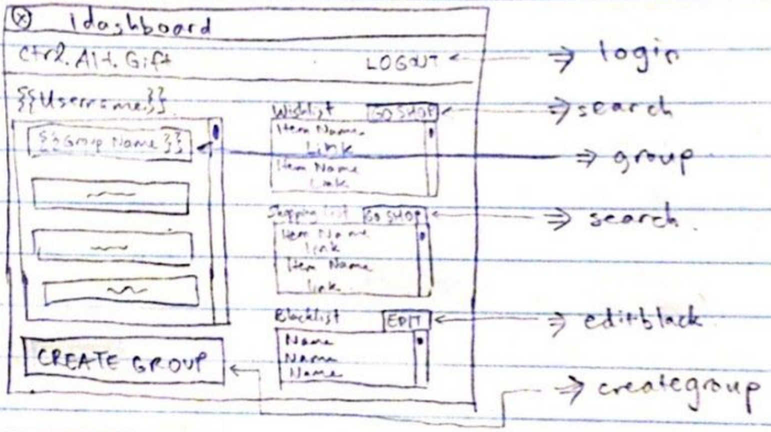
- Accounts
- Groups — Sign up required
- Shuffle
- Wish List
- Pairing Black List
- Shopping List

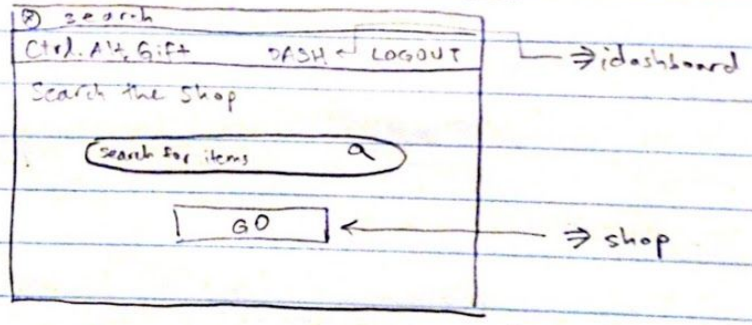
Program Components:

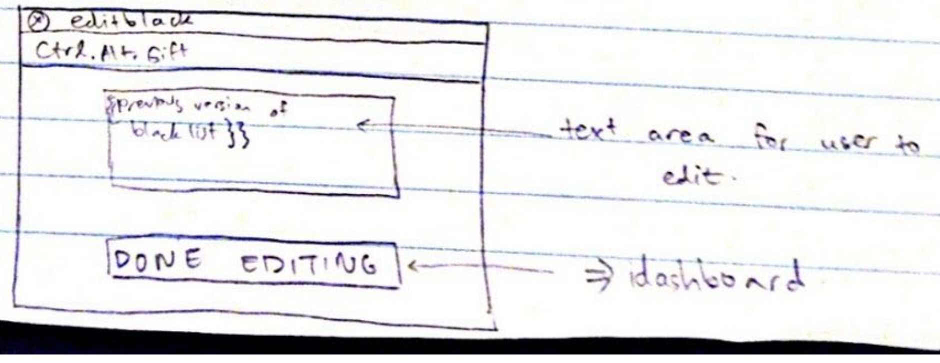
- app.py
- utils/
    - data.py
    - group.py
    - amazon.py
    - etsy.py
    - auth.py
    - data.db
- templates/
    - basic.html
    - login.html
    - group.html
    - search.html
    - shop.html
    - creategroup.html
    - idashboard.html
    - editblack.html
- static/
    - styles/
        - bootstrap/
        - custom.css
    - scripts/
        - script.js

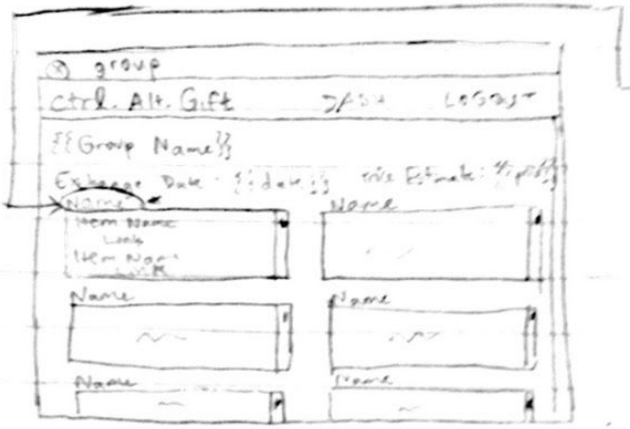
SITE MAP (1)


login
Ctrl. Alt. Gift
Login
Register
Username
Password
Name
SUBMIT
Toggle between login/register ⇒ uses JS to add/remove text input for name.
text input
⇒ idashboard

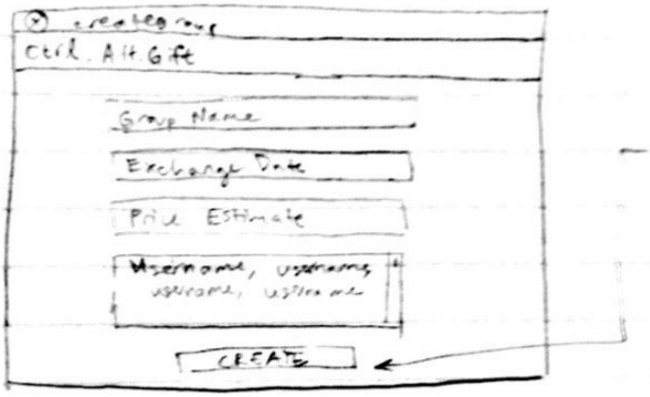

idashboard
Ctrl. Alt. Gift
LOGOUT
{{Username}}
{{Group Name}}
Wishlist
GO SHOP
Item Name
Link
Item Name
Link
Shopping List
GO SHOP
Item Name
Link
Item Name
Link
Blacklist
EDIT
Name
Name
Name
CREATE GROUP
⇒ login
⇒ search
⇒ group
⇒ search
⇒ editblack
⇒ creategroup


search
Ctrl. Alt. Gift
DASH
LOGOUT
Search the Shop
Search for items
GO
⇒ idashboard
⇒ shop

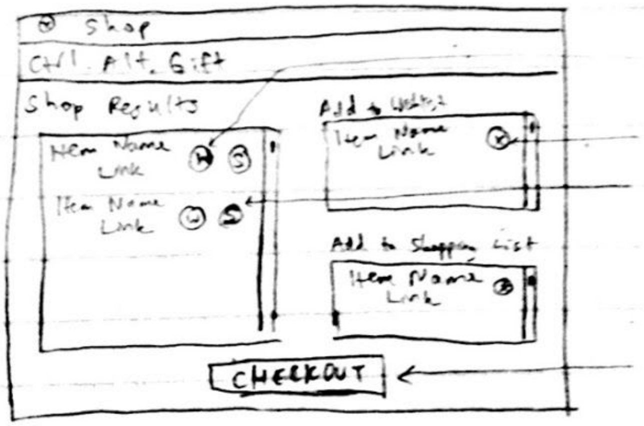

editblack
Ctrl. Alt. Gift
{previous version of black list}}
DONE EDITING
text area for user to edit.
⇒ idashboard



Highlighted to indicate to user who they need to get a gift for



— Creates a group. shuffles members.
⇒ group



JS : add to wishlist

JS : remove item

JS : add to shopping list.

update wishlist and
shopping list

⇒ idashboard

Site Map Description:

- basic.html
    - Other templates will extend this one. Includes the basic headers, links to any necessary stylesheets/scripts.
- login.html
    - Login page. Form with username and password.
    - JS so that a registration page can pop up if the user has not created an account yet
- group.html
    - List of members of a specific group, as well as what each member has on their wishlist. Buttons to leave group (which will then redirect you to idashboard.html) and shuffle (disabled if already shuffled).
- search.html
    - Input for group name, search bar to look for items on Amazon/Etsy.
- shop.html
    - Ues Amazon and/or Etsy APIs to generate a list of related products. Can add items (below price limit) to shopping/wish list. Finished button at the bottom, redirects to idashboard.html.
- creategroup.html
    - Form that takes inputs for members to add, exchange date, price limit. Then redirects to group.html.
- idashboard.html
    - Dashboard for a specific user. Displays username, a list of groups that the user is included in, and the user's lists (black list, wish list, and shopping list).
- editblack.html
    - Form to edit a user's blacklist. Textarea initially set to the original data.

Component Breakdown:

- app.py
    - Includes functions for login/session stuff.
    - Includes Flask routes, etc.
- auth.py
    - Includes all functions for user authentication and hashing
- data.py
    - Includes data and database related functions
- group.py
    - Includes functions for generating groups and matching/shuffling participants
- amazon.py
    - Includes all functions related to the Amazon API
- etsy.py
    - Includes all functions related to the Etsy API
- data.db
    - Contains all user data, user lists, and group information

Database Schema:

- TABLE groups - group composition and shuffling
    - groupID INTEGER - unique group ID, used for organization of groupdata
    - groupName TEXT - name of group
    - username TEXT - usernames of user in group
    - recipient TEXT - usernames of gift recipient
- TABLE groupdata - metadata for each group
    - groupID INTEGER - unique group ID
    - members INTEGER - number of members in group
    - * budgetLow REAL - value of lower price limit
    - * budgetHigh REAL - value of higher price limit
- TABLE userdata - user data and information
    - username TEXT - username of user
    - name TEXT - name of user, ‘First Last’
    - password TEXT - hashed password of user
    - email TEXT - user email address
- TABLE wishlists - a user’s wishlist requests
    - username TEXT - username of user
    - itemName TEXT - name of item
- TABLE blacklists - a user’s blacklist
    - username TEXT - username of user
    - ignoreUser TEXT - username of person to blacklist
    - ignoreName TEXT - name of person to blacklist
- TABLE shoppinglists - a user’s shopping list
    - username TEXT - username of user
    - itemName TEXT - name of item
    - link TEXT - link to item on shopping website

* May not be implemented in final version.

Delegation:

Stephanie: Front end, design/management
Winston: Flask, APIs
Will: Project Manager, Front end (mostly JS)
Zicheng: Databases, Backend

Dev Timeline:

By Thursday, January 5th (evening):

- Will: Create GitHub repository, add contributors. Check in the rest of the group to make sure the dev timeline is acceptable, adjust if necessary. Finalize and submit design document.
- Zicheng: Review design document. Write up database schema.

- Winston: Review design document. Write up component break down.
- Steph: Site map image.

By Sunday, January 8th (evening) :

- Will: List management (define JS functions for how to remove items from a list, add items (for the shopping list, below price range) to a list). Write up README to include abstract and any special instructions.
- Zicheng: Groups (define python functions for how to shuffle the members of a group, avoid matching members on blacklists). Database (organizing data.db, python functions for how to register/login, update black/wish/shopping lists).
- Winston: Flask application (including the registering/login code). Search bar, generating lists of items from Amazon and Etsy below the price limit.
- Steph: Create html templates, Bootstrap and customized styling and create a list of jinja variable names and what information they require.

By Thursday, January 12th (morning):

- **At this point a working version of the site should be completed, from here probable bugs are to be fixed and other features could be added**
- Will: Check in with the rest of the group to make sure everything is completed so far. Adjust design document/code if necessary. Organize test run.
- Zicheng: Check in with the rest of the group to make sure everything is completed so far. Adjust design document/code if necessary. Organize test run.
- Winston: Check in with the rest of the group to make sure everything is completed so far. Adjust design document/code if necessary. Organize test run.
- Steph: Check in with the rest of the group to make sure everything is completed so far. Adjust design document/code if necessary. Organize test run.

By Monday, January 16th (morning):

- Will: Finish. Final test run. Email Mr. DW the API keys if successful.
- Zicheng: Finish.
- Winston: Finish.
- Steph: Finish.