CompSci 316 - Fall 2020

Project Details

Please read carefully and choose one of **"1. Fixed Project Option"** and **"2. Open Project Option"**. You will be asked when the classes start about your intent to do one of these (or if you are okay with doing either option) so that we can form project teams from the same discussion sessions accordingly. Each project must have exactly 5 members unless the total number of students in a discussion session is not a multiple of 5, when we might have to do some adjustments.

1. Fixed Project Option: Build your own "Mini Amazon"

Index:

A sample design of content and actions on each page

# This is a sample design, feel free to have your own design of the webpage as decided by your project team as long as the main features

(buyers, sellers, items, categories, cart, order, reviews etc. and their interactions) are present. See “division of work” for the must-have concepts in the website.

Login page:

- The login page can be implemented after the rest of the website is mostly implemented to avoid delays. We can test the bulk of the website pages with a hardcoded seller account and a hardcoded buyer account, and implement the login page before you test the transactions features at the end.
- Two input fields: email/username and password
    - In case of incorrect input, website should show “incorrect email / password combination”
- Two links: registration and forget password
    - Registration requires the user to input all information to register for a new account
    - Forget password lets user retrieve password by answering the secret question set in registration or by sending an email

Main page:

- The user must be logged in to enter the Main page or else the user will be sent back to the login page
- Search options: search by string/value match on item name, category, price, etc (design/details left to the project groups). Jump to the result page after searching.
- Recommended item list: (design/details left to the project groups) e.g., can recommend the best 3 items in each category. For the recommended item, show image, price, add to cart button, and item name. Item names should link to the item page.
- **User dropdown:** user name and current balance. Drop down contains links to user profile, add balance and purchase history.If user is a seller, add item, modify item and selling list should also appear on the drop down as links to each page (User dropdown will be repeated in several pages).
- **Cart:** link to cart page (Cart will be repeated in several pages).

Search result page:

- Miniamazon icon: on top left as link back to main page.
- User dropdown: (as before)
- Cart: (as before)
- List of items that match the search options: item image, item name, description, and add to cart. The item name should link to the item page.

Item page:

- Miniamazon icon: (as before)

- User dropdown: (as before)
- Cart: (as before)
- Category information for the item.
- Item information: Name, brief item description, image, review/average rating.
- Selling information: A list of sellers selling this item. The list contains seller name, availability, count, price, and add to cart button. Note that different sellers may have different selling prices for the item as well as different numbers of copies available.

Cart page:

- Miniamazon icon: (as before)
- User dropdown: (as before)
- Cart: (as before)
- List of items in the cart: item image, item name, seller name, price, count, and remove button. Item name should be the link to the item page. You can think about allowing the user to edit the cart (e.g., by removing an item or by changing counts with "+"and "-" buttons)
- Total price at bottom: total price - considering all copies of the same item that have been added. For the same item, the user may add it from different sellers?!
- Check out button: successful check out will reduce money from buyer and add money to seller. All items in cart will be recorded in purchase history. If the user has not enough money in his/her account, "insufficient funds" should show up. If the seller has not enough copies to sell, something like "insufficient number of copies of an item, please buy from other sellers" should show up.

User profile page:

- Miniamazon icon: (as before)
- User dropdown: (as before)
- Cart: (as before)
- User profile: all useful user information you can imagine (and a photo too?)!

Add balance page:

- Miniamazon icon: (as before)
- User dropdown: (as before)
- Cart: (as before)
- Shows current balance.
- One input: how much balance you want to add to this account
- One button: proceed.
  - In case of a successful transaction, "success!!" should show up
  - In case of unsuccessful transaction (negative number, not-number chars), "try again" should show up.
- Feel free to update the payment design, e.g., by using a credit card!

Purchase history page:

- Miniamazon icon: (as before)
- User dropdown: (as before)
- Cart: (as before)
- List of items purchased: item image, item name, seller name, price, count and purchased date/time. Item name should link to the item page.
    - Note: the same user may buy the same item at the same/different quantities from the same/different sellers multiple times!

Trade history page (only if the user is also a seller):
- Miniamazon icon: (as before)
- User dropdown: (as before)
- Cart: (as before)
- List of items sold: item image, item name, buyer name, price, count, and date/time. Item name should be the link to the item.

Selling list page (only if the user is also a seller):
- Miniamazon icon: (as before)
- User dropdown: (as before)
- Cart: (as before)
- List of items being sold by a seller: item image, item name, price, and count. Item name should link to the item page. A delete button should appear on the right of each item to delete this item.

Add item, modify item page (only if the user is also a seller, can also be part of selling list page):
- Miniamazon icon: (as before)
- User dropdown: (as before)
- Cart: (as before)
- Input space for all fields needed for an item: item name, item image, item description, seller (this seller only) description, price, and count.
    - “Add item” should have all fields blank
    - “Modify item” should have them filled (editable) with previous values.
- Proceed button: confirms the transaction

Add review page:
- Miniamazon icon: (as before)
- User dropdown: (as before)
- Cart: (as before)
- Input space for rating for item, rating for seller (can be optional), and free text for comments.
- Submit button: confirms the transaction

Step 1: Setup (flask, sql, git, slack/team)

- Everything should already be ready to install by init.sh.
- Additional help is available through Duke Colab (https://colab.duke.edu/).
- If you are coding with vuejs and Flask, additional installation is needed:

```
sudo apt-get install npm
npm install vue   #https://vuejs.org/v2/guide/installation.html
npm install -g @vue/cli
```

- **Do not forget to set up a shared gitlab (https://gitlab.oit.duke.edu/) or github repository and submit your code individually so that the TAs can keep track of who is contributing.**
- Please also develop good coding practices (comments, using meaningful variable names, other documentation, etc.)
- We recommend using Slack or Microsoft Teams for your group collaboration where you can create multiple channels / threads.

Step 2: Designing the E/R diagram

- Design a reasonable E/R diagram based on the website design above (or your design).
- It should have concepts like Users (buyers), Sellers, Items, Categories, Cart, Order/Selling history, Review, Recommendation etc along with their interactions.
- You may add additional features as needed.
- Think about appropriate use of n-ary relationship sets, attributes on relationship sets, and subclass (ISA) and weak entity sets.
- Remember that not all database constraints can be (or should be, for simplicity) captured in E/R diagrams.

Step 3: Designing the database schema and constraints

- Design the database schema, triggers, and constraints (key constraints, foreign key constraints, functional dependencies, CHECK etc.) as needed according to your website design and the E/R diagram.
- Note that some of the constraints NOT captured in the E/R diagram might have to be added to the database design.

Step 4: Loading the database

- You have to use a database management system in the backend of your website. You can use Sqlite3, Postgres, or another database system.
- In Milestone-2, you have to show how a basic version of your website works with a small synthetic dataset (have at least 10 items, 2 categories, 3 buyers, 2 sellers, 20 orders, 10 reviews with arbitrary text and scores in your synthetic dataset).
- In the final submission, you will need to show how your website works with a “Production Dataset”.
- Here is an example small Amazon data we have found online but it contains only “toys” as the items. You can also try to find other public datasets with ecommerce data.
  https://www.kaggle.com/PromptCloudHQ/toy-products-on-amazon
  - Some of the columns in your database might be missing in this toy dataset. You have to generate synthetic values for those columns.
  - It needs to be cleaned up and parsed.
  - Find/generate other products if possible and include them to the database.

# Sqlite3 can bulk insert csv file into database. Please refer to the document https://www.sqlite.org/cli.html

Step 5: Implement the pages

- Note: If the main file for the backend is app.py written with flask and python, then the Html templates for the front end should be in a folder in the same directory as app.py.
- Refer to https://flask.palletsprojects.com/en/1.1.x/tutorial/layout/ for a tutorial.

Step 6: Testing

Test for
- Crashes / bad inputs
- SQL injection attack
- Real time interaction
- Scalability
- Concurrent access of the website by multiple users (transaction)

etc.

Step 7: Make the frontend look nice and/or add new features

Here you show your creativity!

Step 8: Submit the Final Code, Report, and Demo

- Please make sure that your Final Code is available in github with commit information from all members.
- Please make sure that your report is professionally written with separate sections for Introduction, Design (with E/R diagram and database design), Implementation, Challenges, Future Work etc. You also need to add who worked on what components. Using latex for writing the report in latex is highly recommended, otherwise use Word. **You will submit a version of this report at every milestone that you will continue to update, and at the end of the semester you will submit the final report.**
- You will have to record a short video for an overview and demonstration of your website at the end of the semester. More details will be posted later.

Division of Work

- Although at the end, different students would be responsible for different components of the webpage, note that a successful project would critically depend on a strong coordination, planning, and collaboration among the group members.
- Note in the “Grading” section that other than weekly updates, all students get the same score in the project unless we find that someone has not been contributing, then only his/her grade is affected.
- However, our experience says that the groups that did excellent projects had very good group dynamics. Conflicts and lack of commitments in groups can result in a not-so-good learning experience for every team member and can also affect the team’s project grades.
- We encourage you to talk to your teammates at the beginning of the semester about the expectations they have from the project (e.g., meeting the basic requirements vs. doing an amazing project) and their time/effort commitment for the project.
- **See some good practices below.**

	Design (E/R)	Design + load data (db)	Backend/ Frontend (based on our design)
Student1	Buyer + profile	Buyer + profile	Login, register, forget password, profile
Student2	Seller + Sells	Seller + Sells	add/modify item, sellingl list
Student3	Item + Category	Item + Category	Search, search result, home page -- Show the recommended items when the user logs in
Student4	Cart + History	Cart + History	Cart, check out, add balance, sales history, order history
Student5	Review + Recommendation	Review Clean/prepare data	Add review (seller and item), update review, show average
Student6 (only for teams with 6 students) See below*	E.g., Shipping + Warehouse Management	Shipping + Warehouse Management	<List the functionalities you will implement>

* Notes for teams with 6 students (only a few teams will be allowed to have 6 members based on first come-first serve as we do not have total number of students a multiple of 5)

- You have to do more work - in particular you have to implement one more functionality to your mini-Amazon.
- One example listed above is shipping and warehouse management -- feel free to think of another functionality!
- When you write to Sudeepa and Yesenia about whether you can form a 6 member team (please copy all 6 members), please write what else you plan to implement.
- The 6-member teams and their proposed work must be approved by Sudeeepa before you start working on the project.

- **Some good practices for group design and implementation:**
  Here are some tips and advice on how to design/implement the project as a team effort.
  - The team has to make common decisions that everyone agrees on in all stages of design / backend / frontend. Some design/code will be used in multiple pages, so before implementing any of them, discuss together how they will be implemented. E.g., (1) what the UI should look like and what should be the design theme, (2) decide how UI elements will be coded across different pages (e.g., plain HTML or some JavaScript frontend library), (3) what the database schema for the project should be, etc. Some students might have more experience with some of these and they can do the basic design/infrastructure that other students may follow while implementing their pages.
  - All the components interact with each other, and your part might be getting input from another team member. For the backend, you need to develop the API together, discussing what the input (output) format of your part should be, which will be the output of (input to) someone else's part in your team.
  - Try to follow "stub implementation" to develop/test your part without much dependency on other components. For instance, while building your part, you can initially test with a hard-coded value of inputs from others, and later coordinate with your team members to see that the components correctly interact with each other. Similarly, you can have item search always return a fixed list of valid items before the actual implementation is up.
  - The login/authentication page can be hard to figure out. Before you get this part working, develop and test the rest of the code using hardcoded user ids (one for buyer and one for seller). If you still cannot make it work, ask us for help.

- **Some good practices for group collaboration:**
  Here are some tips and advice for group collaboration.

- Have a gitlab (oit) or github repository. It should have a master branch, and then individual branches for testing different students' work. Before pushing changes to the master branch, discuss with your teammates.
- Have a Slack (or Teams) set up to communicate with your team members instead of using emails. Here you can create channels to discuss frontend, backend, design, code, etc.
- Keep updating your report in a shared online document that everyone can edit, e.g., on overleaf.com if you are using latex, and google doc otherwise.
- Schedule a weekly meeting time right away that everyone must attend, where you will discuss what you did last week and what you will do next week. Make sure that you have a plan to meet all the milestones and the final deadline.
- When team members post weekly updates on Monday in your private Piazza threads, make sure that everyone is on the same page and agree with the updates. Better, check these updates within your group (in slack) before these updates are posted.
- If you face a conflict, try to resolve yourselves within the group first and then let the TAs and Yesenia know as soon as possible. We might not be able to do much if you draw our attention to it only at the very end before the final submission. If you see that a team member is missing on his/her deliverables for some time, let your TAs and Yesenia know. We might be able to give you some code to replace his/her work. Make sure that you are checking in your code to git regularly so that the TAs can find out who is contributing.

Submission and Grading

- Project has **20% weight** in the course grading

- **Total 100 (+ up to 15 bonus points for wow factor!)**

- **10 points for weekly updates for all weeks** -- this part is individually graded.
- **The other 90 points + bonus points will be the same for all group members**, **unless a group member does not deliver his/her part of the project.**

- Getting the top grades for projects and bonus points will require a strong coordination and hard work among all the group members!

Milestones (MS):

1. **MS1 (20 points)**
    a. Submit the E/R diagram **(6 points)**

b. Submit the database design (keys and constraints) **(6 points)**
c. A list of assumptions **(2 points)**
d. A description of the web interface by copying and updating the one given by us as per your own design **(6 points)**
e. Mention who is working on what as outlined in the Work Division section (Student 1..5 should be replaced with names of team members).

2. HW-5 (Group assignments -- graded as an assignment)

a. A basic fully working website:
   - Database + One frontend page that interacts with the database :: e.g., can list all items, and items matching a name (search bar) from the frontend

3. MS2 (30 points)

a. Submit a video 3-4 mins showing the status of your website -- how much of the frontend/backend has been implemented **(10 points)**
b. Show (+ submit code) a working website with limited functionality with small / synthetic data. All the components on the current website should interact with each other correctly. **(15 points)**
c. Write a data procurement / cleaning plan for the final website **(5 points)**
d. Mention any change from MS1 report or division of tasks.

4. **Final submission (40 points + up to 15 points for wow factor!)**

a. Final and complete website **(20 points + up to 11 bonus points)**
   - Nice looking frontend **(5 + up to 5 bonus points)**
   - Production dataset **(5 points)**
   - Additional features a plus **(up to 6 bonus points)**
   - All components should work properly **(6 points)**
   - SQL injection attack taken care of **(4 points)**

b. Report **(10 points + up to 2 bonus points)**
c. Demo video **(10 points + up to 2 bonus points)**

2. Open Project Option: Implement your own idea!

- An open project also builds a database-backed website, but the project idea will be your own! See Sakai in a few days for some sample final video submitted by project teams from Spring’20 semester (these videos can only be accessed by students in the class, please keep them private).
- Except your own idea for the website, you will follow the exact same steps as outlined in the Fixed Project Option. The grading will also be the same as outlined in the Fixed Project Option.
- The additional content you have to submit:
    - In MS1: Submit what your website intends to do. You have to give a detailed design on the website as well (similar to what we have specified for the MiniAmazon project), but you will be able to change the design as the project progresses in the later milestones.
    - If you change your design, please include the updated design in the reports of all subsequent milestones.
- If you are planning to form a 6-member team (see the description in Fixed Project, only a few teams will be allowed for balancing in the class based on first come first serve basis), please make sure to have enough work for all 6 members that you have to list in a table like the Fixed Project when you submit your MS1.