WebShop web application - Project requirements

The webshop is a web application where users can buy and sell items. An Item can be on-sale or sold. An item becomes “sold” when it is purchased by another user. For the buyer, the item will show as “purchased”.

There are 2 kinds of users: **registered** and **unregistered**. Unregistered users are anonymous and can only browse or search available items. Registered users are authenticated and can additionally buy, sell and view their own inventory. This is a list of requirements for the webshop.

Pts	Constraints
4p	**1. Project folder (MANDATORY)**: a. The root folder of the project should contain a ***readme.md*** file listing i. Your name and email address ii. Which requirements optional requirements have been implemented iii. How to run the project (install dependencies, start server(s)) b. root folder must contain **requirements.txt** for the backend c. The frontend folder should contain the **package.json** file with all used packages d. The frontend folder should include both **src** and **build** folders
3p	**2. Backend (Mandatory):** a. Backend uses Django, and provides an API b. Django serves JSON to the shop page and HTML to the landing page c. We use SQLite database for this project. ~~NOTE:~~ ~~● The project server is started with ***python manage.py runserver*** without the need to start another frontend server. (optional)~~
2p	**3. Frontend (MANDATORY):** - The web shop should be implemented ~~as a single-page application (SPA)~~ using React. ~~- The web shop should be available at http://localhost:8080/~~
	Functional Requirements
2p	**4. Automatic DB population (MANDATORY):** on the landing page (for grading purposes) there should be a link or button from where any anonymous visitor should be able to automatically populate the DB with 6 users, of which 3 users (i.e. sellers) own 10 items each. **Before each re-population, the DB should be emptied**. The generated users should have the following format: a. Username: testuser#

	b. Password: pass# c. Email address: testuser#@shop.aa After the data is generated the landing page should be updated with a message
3p	**5. Browse (MANDATORY):** Any user can see the list of items for sale. a. The item (graphical) component should have at least this information: i. Title ii. Description iii. Price iv. Date added
3p	**6. Search:** Any user can search for items by **title**. Note: any search action should result in a request to the API.
2p	**7. Create an account (MANDATORY):** Users should be able to create an account by setting a username, password, and email address. *To make evaluation easier do not enable strong password authentication and do not check emails against regular expressions.*
2p	**8. Login (MANDATORY):** a registered user can log in with a username and password
2+1p	**9. Add item (MANDATORY):** an authenticated user can add a new item to sell by providing. a. Title b. Description c. Price The date of creation should be automatically saved on the backend.
3p	**10. Add to cart (MANDATORY):** An authenticated user can select an item for purchase by adding it to the cart. A user (buyer or seller) cannot add to the cart its own items. The item should still be available for other users to search for and add to their cart.
1p	**11. Remove from the cart:** an item can be removed from the cart by the buyer.
3p	**12. Pay:** the buyer sees the list of items to be purchased. When pressing the “PAY” button: a. If the price of an item has changed for any item in the cart, the cart transaction is halted and i. a notification will be shown next to the item, and ii. the displayed price should be updated to the new price. b. If an item is no longer available when the user clicks 'Pay', the whole cart transaction is halted and a notification is shown to the user without removing the item from the cart. The user can manually remove the unavailable items and then Pay. c. On a successful Pay transaction, the status of each item in the cart becomes SOLD. The bought items are listed as the buyer’s item (but they are not available for sale).

3p	**13. Routing:** The Shop page should be implemented as a SPA. One should be able to navigate from the browser (bar) to the following links: a. Shop "/" b. SignUp "/signup" c. Login "/login" d. Edit Account "/account" e. MyItems: "/myitems"
1+1p	**14. Edit Account:** an authenticated user should be able to change the password of the account by providing the old and the new password.
3+1p	**15. Display inventory:** an authenticated user should be able to visualise his/her own items: on sale, sold, and purchased.
2p	**16. Edit item:** the seller of an item can edit the price of the item as long as the item is on sale (available), via the *Edit* button, regardless of the item being added to any other buyers' carts.
	Non Functional - Look and feel
1p	The web pages should look nice and easy to use on regular desktop screens

Total: 40+3 points

- 24 mandatory points → grade 1
- 16 optional points → grade 2-5

Note: All MANDATORY requirements must be satisfied for the project to be graded. The projects not satisfying them will not be graded. All requirements that are not MANDATORY, are OPTIONAL. You decide which ones to implement to get the desired number of points.

Grading table:

Final grade	Percent of max points	Points
0	0-50%	0-20
1	50-60%	20-24
2	60-70%	24-28
3	70-80%	28-32
4	80-90%	32-36
5	90-100%	36-40