2021

PS1 Code Judgement Rubric

ONLY CODE JUDGES SHOULD BE USING THIS VERSION OF THE RUBRIC!

Judge (not shared with students): _____
Judged Team (name or number): _____
Date: _____

Instructions: for each requirement, mark the box that most closely aligns to your impression of the solution given the prompt. Your judgments here should reflect how well the solution satisfies the directions in the problem statement and the team's presentation as well as your own review of the solution's source code. **Aesthetic appeal of a solution is <u>not</u> more important than functionality.** If a solution is not pretty but otherwise meets a requirement, that counts as reasonably satisfying that requirement. Keep your expectations high, teams had **a whole month** to work on this. **DO NOT FEEL OBLIGATED TO AWARD HIGH SCORES!** In the case that a solution attempts to "satisfy" a requirement by making it *look* like it works via the UI/demo code but the required feature is actually non-functional, <u>a score of 0 is appropriate.</u>

Solution Functionality Review [total weight: 100]

Requirement(s) to Reference	Score			
	0	1	2	3
	Missing or Completely Non-functional	Attempted But Does Not Meet My Expectations	Meets My Expectations	Exceeds My Expectations
Requirement 1 Requirement 5	You **must** login before being able to interact with the app or view any memes. **[weight: 2]**			
	Providing random login credentials *does not* allow you to log into the system successfully. **[weight: 2]**			
	Providing the correct login credentials *does* allow you to log into the system successfully. **[weight: 2]**			
	When you are already logged in, you should not be able to create a new account using the same URL/link used to create accounts when you're not logged in. **[weight: 1]**			
	You can create a brand new account via a registration feature; you can login successfully using the account you just created. **[weight: 2]**			

Requirement 2	The Chats view (the page that lists all the private chats or “DMs” between users) is pleasant to look at, elegant, and ergonomic. It is pretty simple to figure out and use. The Chats view has the appropriate components spelled out in the requirement text and provides the functionality spelled out at each bullet point. **[weight: 5]**			
	Two users can converse memes back and forth privately with one another in real time (lag is okay); new memes/messages should appear without having to refresh the page or press any other buttons. You should create two accounts open in two different tabs to test this. **[weight: 3]**			
	Before sending a meme, you can **optionally** choose an expiration time for it. Select a time several seconds into the future. The meme should be visible in chat by both the sender and the receiver up until it expires, at which point it should disappear or be “vanished” *immediately*. When a meme is vanished via expiration date, it no longer shows up in chat between users. **[weight: 3]**			
	Memes appear in a sensible order. Memes show a timestamp of when they were sent. It is always clear who is the sender and who is the receiver in a DM thread. **[weight: 1]**			
	Memes that include images display those images. Memes that only have text display like a text message. If a meme has images and text, both are displayed. **[weight: 2]**			
	You can create/upload a meme by only providing 1) a URL to an image on the internet somewhere, 2) up to 500 characters of text as a description, *and* 3) both #1 and #2 at the same time. Creating new memes is a pleasant experience. **[weight: 3]**			
	You cannot create a meme with a description/text longer than 500 characters. **[weight: 1]**			

	An already sent meme that had no expiration time can still be deleted or “manually vanished” by its owner retroactively. When a meme is manually vanished, it no longer shows up in chat between users. This does not have to be immediate and can require a refresh. **[weight: 2]**			
	It is obvious to both sender and receiver in the UI when a meme has vanished. **[weight: 1]**			
Requirement 3	The Stories view (the page that lists all the memes shared by a user’s **friends**) is pleasant to look at, elegant, and ergonomic. It is pretty simple to figure out and use. The Stories view has the appropriate components spelled out in the requirement text and provides the functionality spelled out at each bullet point. **[weight: 5]**			
	You can create/upload a meme by only providing 1) a URL to an image on the internet somewhere, 2) up to 500 characters of text as a description, *and* 3) both #1 and #2 at the same time. Creating new memes is a pleasant experience. **[weight: 2]**			
	You cannot create a meme with a description/text longer than 500 characters. **[weight: 1]**			
	Memes appear in a sensible order. Memes show a timestamp of when they were sent. **Users can only see memes shared by their friends and themselves.** **[weight: 1]**			
	While memes in this view do not have expiration times, they can still be deleted (“manually vanished”) at any point; after doing this, the meme should no longer appear to the owner or the owner’s friends. **[weight: 1]**			
	You can share a meme that previously appeared in your user’s Stories view. When sharing a meme, it is as if creating a new meme but without having to fill in a URL or any other details. This shared meme appears in your user’s friends’ Stories views (make a second account and add it as a friend to test this). **[weight: 1]**			

Requirement 4	The Notifications view (the page that lists all of your user's notifications) is pleasant to look at, elegant, and ergonomic. It is pretty simple to figure out and use. The Notifications view has the appropriate components spelled out in the requirement text and provides the functionality spelled out at each bullet point. **[weight: 5]**			
Requirement 5	The Auth view (the page that allows you to login and/or register) is pleasant to look at, elegant, and ergonomic. It is pretty simple to figure out and use. The Auth view has the appropriate components spelled out in the requirement text and provides the functionality spelled out at each bullet point. **[weight: 5]**			
	You are required to upload an image from your local computer (**SO NOT A URL**) as your profile picture when creating a new account. You cannot upload a picture that is too big (larger than 10MB). **[weight: 4]**			
	Registration requires a CAPTCHA. Users are restricted from logging in for 5 minutes after 3 failed login attempts. Users can see how many attempts they have remaining. **[weight: 2]**			
	When logging in, you can use "remember me" functionality to stay logged in. That is: you can login using "remember me," close the browser window, reopen it, and still be logged in afterwards. When you disable "remember me" functionality and you close and reopen the browser the same way, you should no longer be logged in. **[weight: 2]**			
Requirement 6 Requirement 4	You can easily send each other friend requests, view sent and received friend requests, and respond to received friend requests. Once a request is sent, the other user can accept or reject it. If accepted, the two users become friends. If rejected, they do not. Until users are friends, they cannot see each other's shared memes in the Stories view. If they become friends, they can then see each other's shared memes (including past shared memes) via the Stories view. **[weight: 5]**			
	You are notified via the Notifications view when you receive a friend request. **[weight: 2]**			
Requirement 7	You can use the password recovery feature to regain access to an account without the			

	password. This should be available through a “forgot password” link of some kind when visiting the Auth view. **[weight: 2]**			
Requirement 8 Requirement 4	When creating a new meme in either the Stories or Chats views, you can use *mentions* to notify (via the Notifications view) other users in the system about the meme. **[weight: 5]**			
	Mentions become links in the UI that allow you to friend/unfriend the mentioned user and/or view their profile picture and username at least. **[weight: 2]**			
	When creating a new meme in either the Stories or Chats views, you can use *hashtags*. Hashtags become links in the UI that allow you to easily search for all memes in the system using that hashtag (excluding vanished memes and only returning memes you have permission to view from the Stories or Chats view). **[weight: 5]**			
Requirement 9	You can search for memes in the system pursuant to the criteria listed in the problem statement text. **Search results must never include memes your user does not have permission to see**. That is: only memes that would appear in the user’s Stories and/or Chats view should appear via search. Other people’s private memes, or any vanished memes, should **never** appear! **[weight: 5]**			
Requirement 10	The like counts of and comments under memes in the Stories view updates “automatically” when changed in the backend without you having to press a refresh button. This can take many forms including “delayed automatic updates,” i.e. updating a meme’s metadata only after you interact with it (this is what Twitter and Instagram do). You can test this by creating multiple users who view and like the same memes, or looking for pre-created accounts with simulated ongoing activity. **[weight: 4]**			
	Chatting back and forth between two users in the Chats view feels natural. You do not have to refresh the page to see new messages in this view. New messages/memes will appear without a page refresh or having to press any sort of update or refresh button. Chatting back and forth should feel normal, like sending SMS text messages on your phone. **[weight: 1]**			

Requirement 12	When viewing lists of things, be it lists of memes, lists of users, lists of friends, lists of search results, etc., all of these results are *paginated* in an aesthetically pleasing way. See the text of the problem statement for more details. **[weight: 1]**			
Requirement 14	The app fails gracefully when errors happen. You don't lose all your progress in the registration form when you press submit with invalid data, for instance. When loading a lot of data, the app does not freeze or present a broken UI but instead shows a spinner icon or some other loading indicator while waiting. You are never waiting too long for something to happen in response to user input. The app never "breaks," never displays internal error messages or warnings anywhere on the page, the backend server hosting the solution never crashes, never any HTTP 500 errors, etc. **[weight: 1]**			
Requirement 15	The app looks good when you view it on a small smartphone-sized screen. **To simulate this, use your mouse to drag the right edge of the browser window left until it cannot shrink anymore. This shrunken column represents the size of a phone screen.** Is the app still usable in this state? Is it visually pleasant? **[weight: 2]**			
	The app looks good when you view it on a medium tablet-sized screen. **To simulate this, make your browser window half the size of your screen (try windows key + left arrow). This simulates the size of a tablet screen.** Is the app still usable? Is it visually pleasant? **[weight: 1]**			
	The app looks good when you view it on a normal desktop/laptop screen. **Ensure the browser window is maximized. [weight: 2]**			
	Overall, you feel this solution is easy to use. **[weight: 1]**			
	Overall, you feel this solution is beautiful to look at. **[weight: 1]**			
	Overall, you feel this solution functioned well. App functionality that worked fine when the browser window was maximized did not suddenly break when you shrunk the browser to simulate a smartphone or a tablet. **[weight: 1]**			

Instructions: for each of the five criteria below, mark the box that most closely aligns to your impression of the solution's source code after your review. Does their source code meet your expectations of quality? **If you notice source code that seems strangely out of place, overly complex, extremely nonsensical, suspiciously unformatted or very badly organized syntactically, or you otherwise infer that students copy-pasted/plagiarized part of their solution's source code from the internet or some illegal source, inform the chief judge immediately.**

Solution Source Code Review [total weight: 100]

Criterion	Score			
	0	1	2	3
	Does Not Meet Expectations	Partially Meets	Meets Expectations	Exceeds Expectations
Naming Conventions	Variables should be well-named (e.g. **accountSum** is good while **acsu** is bad). There should be few "magic numbers" floating around. The chosen variable naming scheme should be consistent and self-documenting. **[weight: 5]**			
Organization	The code should be well-organized, with a consistent logical file/directory layout, encapsulation, and proper separation of concerns. Similar units of code should be grouped together in a logical way. If object-oriented language features are used, they should be used properly (i.e. SOLID). **[weight: 30]**			
Maintainability and Extensibility	The solution should be implemented such that a future group of programmers would not have to rewrite large portions of the source code to add new functionality. Source code should be DRY where reasonable. Pay special care to inflexible and hardcoded values. **[weight: 25]**			
Readability	How easily readable is the source code? Did you have a hard time going through it? Solution code should be as intuitive and self-documenting as possible. Confusing sections should be documented with comments. There should be few "WTF?!" moments. **[weight: 25]**			
Efficiency	Any implementations should not be too inefficient. Does some piece of code waste cycles looping or otherwise doing something unnecessary? Is the application wasteful with system resources? Does it make more network requests than necessary? **[weight: 15]**			

Positive Comments

Please share any positive comments you have for this team.

Constructive Criticism

Please share any unanswered concerns or comments you have for this team.

Two Questions For This Team

Please share the two most pressing questions you'd like answered about this team's solution.

Shared Thoughts (visible to all judges)

Please add any thoughts or concerns you believe other judges should be made aware of.