Vulnerablity Report For (Demoblaze.com UPDATED *)

Vuln. Name	Severity	Description	POC	Mitigation	Location
API Dierctory Enumeration *	HIGH	API Directory Enumeration are these when a Developer of that Company Poorly Congfigured An API And these can lead to Fatal Attacks as we all know that API's are the Backbone of the Systems	https://youtu.be/dKpcZdonP0	A Develper Should never alow the users to show their Correct API Directory's it can leak out special hidden API's direcories that An attacker should be allowed to see.	Seen on api endpoint
Weak Credientials	HIGH	A weak password is short, common, a system default , or something that could be rapidly guessed by executing a brute force attack using a subset of all possible passwords, such as words in the dictionary, proper names, words based on the user name or common variations on these themes.	https://youtu.be/otSFoRQjMc	Enforce strong password requirements on users Even when credentials are stored using secure hashing algorithms, a weak ord can still be a culprit and an attacker may be able to these passwords using a variety of password cracking techniques such as dictionaries and rainbow tables.	Seen on /login endpoint
No Rate Limit on Signup Form	MEDIUM	This vulnerability allows for user enumeration, and attackers may utilize email and SMS services to launch flooding attacks.	https://youtu.be/vNDcfaIzOnw	To resolve this concern, developers should set a timeout after a certai n number of requests in a given period of time, or use a CAPTCHA system on form pages.	Seen on /Signup Endpoint
UUID Based IDOR	LOW	When looking for IDORs, not only are numeric IDs susceptible, but in some cases Universal Unique Identifiers (UUIDs). A UUID is a cryptographically generated identifier, used in a similar way to IDs, but less vulnerable to enumeration. That said, there is a way to find vulnerabilities.	https://youtu.be/rjwi8-Q9Nas	To Resolve his issue Developer need to chain the UUID with the user Cookies so they can only delete their own cart items	Seen on /deleteitem Endpoint
CORS	MEDIUM	CORS (Cross-Origin Resource Sharing) defines a mechanism to enable client-side cross-origin requests. This application is using CORS in an insecure way. The web application fails to properly validate the Origin header (check Details section for more information) and returns the header Access-Control-Allow-Credentials: true. In this configuration any website can issue requests made with user credentials and read the responses to these requests. Trusting arbitrary origins effectively disables the e-origin policy, allowing two-way interaction by third-party web s	https://youtu.be/7kaopOIkOjA	Allow only selected, trusted domains in the Access-Control-Allow-Origin header.	Seen on /check endpoint
Outdated JQUERY	MEDIUM	this site is using an outdated version of Jquery libraries. A more recent version is available. This version was found to be affected by Some security vulnerabilities, it is recommended to keep libraries up to date.	https://youtu.be/4K7nwyK6-HM		
Imporper Input Validation	MEDIUM	Input validation is performed to ensure only properly formed data is entering the workflow in an information system, preventing malformed data from persisting in the database and triggering malfunctionof various wnstream components. Input validation should happen as early a possible in the data flow, preferably as soon as the data is received from the external party.	https://youtu.be/aDMxv1hoSdg	It is always recommended to prevent attacks as early as possible in the processing of the user's (attacker's) request. Input validation can be used to detect unauthorized input before it is processed by the application.	Seen on /cart.html By placing order
Username Enumeration	LOW	It may be possible to enumerate usernames, based on differing HTTP responses when valid and invalid usernames are provided. This would greatly increase the probability of success of password brute-forcing attacks against the system. Note that false positives may sometimes be minimised y increasing the ‘Attack Strength’ Option in ZAP. Please manually check the ‘Other Info’ field to confirm if this is actually an issue.	https://youtu.be/jlcNZjrDfO0	Do not divulge details of whether a username is valid or invalid. In particular, for unsuccessful login attempts, do not differentiate between an invalid user and an invalid password in the error message, page title, page contents, HTTP headers, or redirection logic.	Seen on /login

Vulnerability Report For (petstore.octoperf.com)

Stored Xss on signup	HIGH	Stored XSS, also known as Type-1 or Persistent XSS attacks, typically rely on unsanitized user input points for scripts permanently stored on the target servers.	https://youtu.be/C9yVk3aGduc	Apply context-dependent encoding and/or validation to user input rendered on a page	Seen on /Signup
Improper Session Handling	MEDIUM	This Site is not applying Any changes after Re-Login It's keep reseting to Values that are used in Signup Process	https://youtu.be/gK3UxrbMkhc	kindly Fix the Cookie issues in your WebApp	Seen on /editAccountForm
Session Token in URL	MEDIUM	This WebApp is storing cookies in JSESSIONID value but once i'm specifying JSESSIONID value in url nowi can delete cookies from firefox dev tools and i can still be logged in	https://youtu.be/M_XkuDt5MPU	Attacker Can intercept the Victim's network and can get this JSESSIONID value. If attacker can gain this victim session ID it's an Account Takeover. So Not passing the Session ID in url is the best security emplementation	Seen on the URL
Improper Input Validation	MEDIUM	Input validation is performed to ensure only properly formed data is entering the workflow in an information system, preventing malformed data from persisting in the database and triggering malfunctionof various downstream components. Input validation should happen as early as possible in the data flow, preferably as soon as the data is received from the external party.	https://youtu.be/wvASRVbgjAc	It is always recommended to prevent attacks as early as possible in the processing of the user's (attacker's) request. Input validation can be used to detect unauthorized input before it is processed by the application.	Seen on /Signup