Major Feature	Issue / Ticket ID	Title	Description	Purpose	Requirements	Expected Outcome	Checklist	Labels	Labels 2
Authentication (AUTH)	BE-AUTH-001	Create Registration Form	Create a user-friendly registration form	To allow new users fill in their details to sign up for the application	Form input fields for: - Full Name - Email Address - Phone Number - Confirm Password - Submit button	- User should be able to fill the necessary fields	- Is the frontend registration form UI created? - Are the required fields included?	Frontend	
Authentication (AUTH)	BE-AUTH-004	Create the Login Form			To allow returning users fill in their details to log in through OAuth into the application	Input fields for: - Username / Email Address - Password - Google OAuth - Facebook - GAuth	- User should be able to fill the necessary fields, or if it's a new user, that user will be redirected to the registration form	- Is the login form UI created? - Are the required fields obvious to the user? - Is the client-side validation implemented? - Are new users redirected to the registration form?	Frontend	
Authentication (AUTH)	BE-AUTH-003	Validate user inputs	Validate the entries the user inputs in the fields	To ensure the user input is secure and valid	- Develop validation logic for user-submitted data - Validate username/email format (e.g., email address must be a valid format) - Ensure password meets complexity requirements (e.g., minimum length, character types) - Implement confirmation password field validation to ensure passwords match - Display clear error messages for invalid inputs	- The user input data should be successfully sent to the backend, or the user will receive an error message if unsuccessful	- Is the user's email format correct; if not, did the user receive an error message? - Do the Passwords sent by the user meet security requirements? - Does the user's Password meet the security criteria (e.g., minimum length, character types)?	Backend	
Authentication (AUTH)	BE-AUTH-002	Implement Registration Form	Connect the registration form to the backend (database)	To ensure the user's response are stored in the database	Corresponding database fields for: - Full Name - Email Address - Password - Confirm Password - Hash user password	- The data entered by the user should be sent to the backend for processing and storage	- Is the server-side validation working? - Are the user credentials securely stored? - Is the user's password in the database hashed?	Backend	
Authentication (AUTH)	BE-AUTH-005	Implement Login Form	Link the login form to the backend for server-side validation. Notify users of login attempts.	To ensure the user securely accesses his/her profile (an authorized page). Enhance security with login notifications.	Corresponding database fields for: - Username / Email Address - Password - Verify email address is unique in the database - Send notifications for login attempts	- The user should be redirected to the subsequent page if successful, or receive an error message if unsuccessful - The user should receive a login notification	- Is the server-side validation triggered? - Are the user's credentials transmitted securely to the Backend? - Notification setup, Email/SMS integration	Backend	
Authentication (AUTH)	BE-AUTH-007	Implement Password Reset	Reset forgotten passwords by sending a password reset link to their registered email address.	Allow users to reset their forgotten passwords	- Button / Link for Forgot Password - Notification for user to check their email for a reset link - Verify user exists - Encrypt new password	- User should immediately receive a password reset link once the request is made	- Was it verified that the user actually exists for a record? - Was the newly created password hashed?	Backend, Frontend	
Authentication (AUTH)	BE-AUTH-006	Implement Two-Factor Authentication (2FA) *Optional*	Require a secondary verification step beyond the password, such as a code sent via SMS or generated by an authentication app	To offer an extra layer of security	- TOTP (Time-based One-Time Password) - SMS Delivery (messaging feature)	- User should be asked for a TOTP during login	- Was a secondary library selected? - Has session management been implemented? - Has SMS delivery and receipt been tested?	Backend	
Authentication (AUTH)	BE-AUTH-008	Session management	Review and improve how sessions are managed, including expiration policies, token handling, and session hijacking prevention.	Securely authenticate users, maintain their stateful interactions, and protect against unauthorized access	- description of current session management - identification of issues or vulnerabilities - security considerations - impact on user experience - testing and validation	- The user should have an active session	- Was a session successfully established and maintained? - Did the user receive some sort of feedback that his/her session is active?	Backend	
Authentication (AUTH)	BE-AUTH-011	Implement reCAPTCHA verification in Registration Form	Add reCAPTCHA verification to the registration form to prevent automated sign-ups	To enhance security by preventing automated bots from creating fake accounts through the registration form	- Integrate reCAPTCHA verification into the registration form - Validate reCAPTCHA responses before allowing form submission	- The registration form should be protected against automated sign-ups using reCAPTCHA	- Is the reCAPTCHA verification into registration form integrated? - Is the reCAPTCHA response validated before form submission? - Handling potential error messages	Backend	Frontend
Authentication (AUTH)	BE-AUTH-012	Implement reCAPTCHA verification in Login Form	Add reCAPTCHA verification to the login form to prevent automated login attempts.	To enhance security by preventing automated bots from attempting to log in to user accounts	- Integrate reCAPTCHA verification into the login form - Validate reCAPTCHA responses before allowing form submission	- The registration form should be protected against automated login using reCAPTCHA	- Is the reCAPTCHA verification into login form integrated? - Is the reCAPTCHA response on form submission validated?	Backend	Frontend
Authentication (AUTH)	BE-AUTH-009	Implement Account Activation via Email	Send an account activation link to the user's email upon registration	To ensure that users verify their email addresses and activate their accounts securely	- Generate a unique activation link for each new user - Send the activation link to the user's email - Create an API endpoint to handle activation link verification and activate the account	- Users should receive an activation email and be able to activate their account via the link	- Is a unique activation link generated? - Is the activation link sent via email? - Is the API endpoint that verifies the activation link and activates the account created?	Backend	
Authentication (AUTH)	BE-AUTH-010	Create Backend API Endpoint for Password Recovery	Develop the backend API endpoint to handle password recovery requests.	To securely process password recovery requests and send password reset emails to users.	- Create an API endpoint to accept password recovery data - Validate incoming data for email format - Send a password recovery email with a reset link	- The backend should securely handle password recovery requests and send recovery emails	- Backend API Endpoint for Password Recovery created	Backend	
Authentication (AUTH)	BE-AUTH-013	Implement Google Login Button	Add a Google login button for social authentication.	To provide users with the option to log in using their Google accounts, offering a convenient and secure authentication method.	- Design and implement a Google login button on the login form - Integrate Google OAuth for authentication.	- Users should be able to log in using their Google accounts.	- Google OAuth for authentication implemented - Authenticated user maps the inputs created - Backend logic for OAuth authentication implemented	Backend	
Authentication (AUTH)	BE-AUTH-016	Create Backend API Endpoint for Verify Magic Link	Develop the backend API endpoint to verify magic login links and log them into the application.	To securely authenticate users using a magic link and log them into the application.	- Create an API endpoint to verify the magic link - Validate the magic link and authenticate the user - Log the user in upon successful verification.	- The backend should securely verify magic links and log users in	- API endpoint for magic link verification - Magic link validated - Authenticate user and log them in	Backend	
Authentication (AUTH)	BE-AUTH-019	Idle Session Timeout		Automatically log out users after inactivity.	Enhance security with session timeouts.	Configure session timeout settings.	Users are logged out after inactivity	Implement session handling with unique token IDs. Ensure sessions are stored securely	Backend	
Authentication (AUTH)	BE-AUTH-022	Multi-device Login Support	Allow users to log in from multiple devices.	Enhance user convenience with multi-device support.	Implement multi-device session management	Users can log in from multiple devices.	Session management, Device tracking	Backend	
Authentication (AUTH)	BE-AUTH-020	Secure Password Storage	Implement secure password storage techniques	Enhance security with secure storage.	Configure secure password storage methods (e.g., hashing, salting).	Passwords are stored securely	Storage setup, Security configuration	Backend	
Authentication (AUTH)	BE-AUTH-021	Recovery Link Expiration	Set expiration for account recovery links	Enhance security by expiring old links.	Configure link expiration settings.	Recovery links expire after a set period	Update password reset functionality to handle expired links	Backend	
Authentication (AUTH)	BE-AUTH-023	Multiple Recovery Options	Provide multiple account recovery options	Enhance user experience with varied recovery methods and increase account security	Implement options like email, SMS, and security questions.	Users can recover accounts using preferred methods.	Recovery setup, Option configuration	Backend	

Authentication (AUTH)	BE-AUTH-015	Implement "Remember Me" Functionality	Implement an optional feature that allows users to stay logged in for a specified period	This increases the need for a user to authenticate each time	Sessions should be created for password reset, and there is a session on the backend.	- User should stay logged in for the specified period	- Is the user's session created?	Backend	
Authentication (AUTH)	BE-AUTH-014	Implement Facebook Login Button	Add a Facebook login button for social authentication	To provide users with the option to log in using their Facebook accounts, offering a convenient and secure authentication method	- Design and implement a Facebook login button on the login form - Integrate Facebook OAuth for authentication	- Users should be able to log in using their Facebook accounts	- Design and implement a Facebook login button - Facebook login implemented - Backend logic for OAuth authentication implemented	Backend	
Authentication (AUTH)	BE-AUTH-017	Create Backend API Endpoint for Magic Link Generation	Develop the backend API endpoint to generate and send magic login links	Securely create and send a magic login link to users to log in without a password by using a magic link sent to their email	- Create an API endpoint to generate magic login links - Generate a unique login link and send it to the user's email	- The backend should securely generate and send magic login links	- API endpoint for magic link request - Unique login link generated - Magic login link sent to user's email	Backend	
Authentication (AUTH)	BE-AUTH-018	Login Throttling	Implement account lockout after too many failed login attempts	Enhance security by limiting the number of failed login attempts to prevent brute-force attacks	Configure account lockout after multiple failed login attempts	Accounts are locked after repeated failed login attempts	- Login attempt tracking implemented - Account lockout functionality tested	Backend	
Authentication (AUTH)	BE-AUTH-024	Implement Email Notification System	Send automated email notifications to users	Keep users informed about important events and updates	- Design and implement an email notification system - Personalize email content where necessary	- Define notification triggers (e.g., account creation, password reset) - Ensure delivery and handling of retries	- Users receive timely and relevant email notifications	- Design email notification system - Define notification triggers - Implement retry handling - Delivery and handling tested	Backend	

Major Feature	Issue / Ticket ID	Title	Description	Purpose	Requirements	Expected Outcome	Checklist / Acceptance Criteria	Labels
Authentication (AUTH)	FE-AUTH-001	Implement User Sign Up page	Implement the user registration flow	To enable users to easily sign up on the application	Implement the user registration flow with the following form Input fields: - Full Name - Email Address - Password - Submit button -Implement specific errors for all fields.	- The registration form includes fields for full name, email, password. - Implementation adheres to the platform's branding guidelines. - Upon successful input of valid data, a success message is displayed; otherwise, an error message appears. - Users should receive clear, actionable feedback for errors	1. The form must include fields for full name, email, and password as well as a "Sign Up" button, a "Log In" link for existing users, and social signup buttons. 2. The implementation of the form must adhere to the platform's branding guidelines. 3. Upon successful input of valid data in all fields, a success message must be displayed to the user. 4. An error message must appear if the user inputs invalid data in any field and should the password strength meter. 5. Implement specific errors for all fields.	Frontend
Authentication (AUTH)	FE-AUTH-002	Implement User Login Page	Implement the user login flow	To allow users to securely log in and manage their accounts.	Implement the user login flow to include Input fields for: - Username / Email Address - Password - Google - Facebook - Magic Link - Remember me -Implement specific errors for all fields.	- Implementation should make users to be able to fill the necessary fields, or if it's a new user, that user will be redirected to the registration form. -Users should receive clear, actionable feedback for errors	1. The login form must include email/username and password fields, a "Forgot Password?" link, a "Log In" button, and a "Sign Up" link. 2. Appropriate messages must be displayed: a success message for valid data input and an error message for invalid data input. 3. Social signup buttons must be present. 4. Remember me check box must be present. 5. Magic link button should be available. 6. Implement specific errors for all fields.	Frontend
Authentication (AUTH)	FE-AUTH-003	Implement Forgot Password Page	Implement the Forgot Password flow	To allow users to reset their password if forgotten.	- Implement Forgot Password form with fields for: - Email Address - Send OTP -New Password - Validate email format. - Ensure new password meets complexity requirements (e.g., minimum 8 characters, uppercase, lowercase, number, special character). - Display clear error messages for invalid inputs (specific to email format, password complexity, and password confirmation). - Hash user password before submission for security. -Implement specific errors for all fields.	- The email address field is included. - A "Send Reset Link" button is added. - Implementation adheres to the platform's branding guidelines. -Users should receive clear, actionable feedback for errors	1. Ensure that the email address field is included. 2. Add a "Send Reset Link" button. 3. Include a "Log In" link. 4. Adhere to the platform's branding guidelines.	Frontend
Authentication (AUTH)	FE-AUTH-004	Implement Logout Functionality	Implement the users logout flow	To allow users to log out of their accounts.	Implement the users logout flow	- Implementation includes a logout button accessible from "My Account" drop down. - Confirmation prompt asks the user if they are sure they want to log out. - Upon successful logout, the user is redirected to the landing page. -Users should receive clear, actionable feedback for errors	1. UI should include a logout button that is easily accessible from "My Account" drop down in the application. 2. UI should include a confirmation prompt asking the user if they are sure they want to log out. 3. Upon successful logout, the user should be redirected to the landing page.	Frontend
Authentication (AUTH)	FE-AUTH-005	Implement Account Deletion	Implement the account deletion flow	To allow users to delete their account	- Implement "Delete Account" button with a confirmation modal and optional reason input; update UI to reflect deleted status. - Provide loading indicators and success/error messages throughout the deletion process.	- Users can initiate account deletion with clear explanations of consequences. - Confirmation options for deletion. - Clear instructions for deleted users. - Error messages for failed deletion attempts or network errors.	1. Users can initiate the account deletion process, clearly explain the consequences of deleting the account. 2. After initiating the deletion, the user should be able to confirm their decision, include options for the user to cancel the deletion if desired. 3. Users should be able to confirm that their account has been deleted, there should be clear instructions on what actions. 4. Error messages for scenarios such as failed deleted attempts or network errors, ensure users are informed of any issues that prevent account deleted.	Frontend
Authentication (AUTH)	FE-AUTH-006	Implement Error Handling	Implementation to address error handling for potential issues.	To allow users to have clear feedback incase of errors	-Implement a global error handler to catch unexpected errors and display user-friendly messages. -Validate form inputs and display specific, clear error messages for invalid fields. -Handle API response errors by providing appropriate user feedback and differentiating between client-side and server-side errors. -Display loading indicators during actions and show success messages or retry options for recoverable errors.	Users should receive clear, actionable feedback for errors, enhancing their overall experience and interaction with the application.	1. The error message should be user friendly and written in clear, consise language. 2. The system should avoid generic error messages like "verification failed" 3. Visual cues can be implemented to guide users towards successful completion of the challenge.	Frontend

Ticket #	Feature	Task Description	Assignee	Acceptance Criteria	User Story
1	Design System	Ensure all the designs are in sync with the system provided	@ife_dagra, @bel	The design must adhere to the guidelines provided in the design system.	As a user, I want all the screens to be consistent so that I don't get confused
2	User Registration Page Design	Design the user registration flow	@web3deus @Folowosele Funke - (Desktop, Mobile, Tablet)	1. The form must include fields for full name, email, password, and password confirmation, as well as a "Sign Up" button, a "Log In" link for existing users, and social signup buttons. 2. The design of the form must adhere to the platform's branding guidelines. 3. Upon successful input of valid data in all fields, a success message must be displayed to the user. 4. An error message must appear if the user inputs an invalid data in any field. 5. The password strength meter should be shown. 6. A checkbox agreeing to the terms and conditions/privacy policy should be implemented	As a new user, I want to easily sign up so that I can access the platform's features.
3	User Login Page Design	Design the user login flow	@Muklab, @DesignBroke - (Desktop, Tablet, Mobile)	1. The login form must include email/username and password fields, a "Forgot Password?" link, a "log in" button, and a "Sign Up" link. 2. Appropriate messages must be displayed: A success message for valid data input and an error message for invalid data input. 3. Social signup buttons must be present. 4. A link for Terms of use and privacy policy should be displayed	As a user, I want to securely log in and manage my account.
4	Forgot Password Page Design	Design the "Forgot Password" flow	@Elisa (Desktop, Tablet, Mobile)	1. Ensure that the email address field is included. 2. Add a "Send Reset Password" button. 3. Include a "Log In" link. 4. Adhere to the platform's branding guidelines.	As a user, I want to reset my password if I forget it so that I can regain access to my account.
5	Logout functionality Design	Design the users logout flow	@iamfavoureseosa, @ayo_sam (Desktop, Tablet, Mobile)	1. Design should include a logout button that is easily accessible from any page in the application. 2. Design should include a confirmation prompt asking the user if they are sure they want to log out. 3. Upon successful logout, the user should be redirected to the login page with a confirmation message.	As a user I want to be able to log out of my account
6	Account deactivation/rea ctivation	Design the account deactivation/reactiv ation flow	@Evelyn @Emmanuel Nwoye (Desktop, Tablet, Mobile)	1. Users can initiate the account deactivation process, clearly explaining the consequences of deactivating the account. 2. After initiating the deactivation, the user should be able to confirm their decision, include options for the user to cancel the deactivation if desired. 3. Users should be able to confirm that their account has been deactivated, there should be clear instructions on what actions, if any, the user can take next (e.g., contact support for reactivation). 4. Error messages for scenarios such as failed deactivation attempts or network errors, ensure users are informed of any issues that prevent account deactivation.	As a user I want to effortlessly deactivate or activate my account
7	Error handling	This ticket addresses error handling for potential issues encountered during reCAPTCHA (or we consider Honeypot for a better user experience)	@Moses.us, @Nifemi that guy (Desktop, Tablet, Phone)	1. The error message should be user friendly and written in clear, concise language. 2. The system should avoid generic error messages like "Verification failed" 3. Visual cues can be implemented to guide users towards successful completion of the challenge.	As a user I want the system to provide clear feedback incase of errors so I can understand the issue and try again successfully.
8	Organization Creation	This ticket addresses creation of organizations. Users can create organizations after authentication.		1. The form must include Company's name, email address and password. 2. An error message must appear if the user inputs an invalid data in any field. 3. The password strength meter should be shown. 4. A checkbox agreeing to the terms and conditions/privacy policy should be implemented.	As a user, I want to be able to create my company's account so that I can collaborate with my team.
8	Password Reset Page Design	Design the password reset flow	@Prince @Temiloluwa Bajomo (Desktop, Tablet, Mobile)	1. The form includes fields for both "New Password" and "Confirm Password." 2. The form includes a "Reset Password" button. 3. The form adheres to the platform's branding guidelines. 4. If we are implementing an OTP, create a page for the user to input the OTP (Optional)	Settings ticket: As a user, I want to set a new password after resetting it so that I can securely access my account.
9	Session Expiration	Design the session expiration flow	@Olumide Adefifa - (Desktop, Tablet, Mobile).	1. When a user's session expires, the system must display an expiration page informing the user of the session expiration and providing clear instructions or messages about it. 2. The expiration page must include an option for the user to log in again and should maintain a consistent design with the rest of the application. 3. Upon selecting the log-in option, the user must be redirected to the login page.	Settings ticket: As a user, I want a seamless re-authentication process so that I can easily login.

ID	Title	Description	Purpose	Implementation Details	User Story	Acceptance Criteria	Tags
AUTH-003	Enable Link-Based Access	Implement functionality to allow users to access the app via shared links without logging in or having an account	To provide seamless access for users through shared links	Implementation Details: - Generate unique, secure access links - Implement access control via these link - Ensure links are time-bound or have a c - Provide fallback for expired or invalid lin	- Users should be able to access shared content without logging in	- Unique, secure access links generated - Access control implemented via links - Expired or invalid links handled appropr	Backend, Security
AUTH-004	Monitor Link Usage	-Track and analyze the usage of shared access links -To ensure the security and effectiveness of link-based access		Implementation Details: - Implement logging for link access events - Store information such as access time, IP address, and user agent - Create reports or dashboards to visualise link usage data - Set up alerts for unusual activity	- Admins should be able to monitor and a	- Link access events logged with relevant details - Reports/dashboards available for usage data - Alerts configured for unusual activity	Monitoring, Analytics
AUTH-005	Expire Access Links	Implement expiration mechanism for shared access links	To enhance security by ensuring links are not indefinitely valid	Implementation Details: - Define expiration criteria (e.g., time-bas - Implement backend logic to check and i - Notify users when a link is about to expi	- Expired links should not grant access to the app	- Expiration criteria defined and documen - Backend logic for link expiration implem - User notification system in place	Security, User Experience
AUTH-006	User Feedback for Link Acce	Collect user feedback on the experience of acc	To improve the user experience and identify potential issues	Implementation Details: - Implement feedback forms or surveys accessible after using a shared link - Store and analyze feedback data - Address common issues or suggestions	- Users should be able to provide feedback on their experience	- Feedback form/survey implemented - Feedback data collected and analyzed - Actionable insights derived from feedba	User Experience, Analytics

Ticket #	Feature	Task Description	Acceptance Criteria	User Story	Boilerplate
2	**User Registration Page Design**	Design the user registration flow	Messaging tickets should include sending emails to users upon updates(e.g to terms and policies)	As a new user, I want to easily sign up so that I can access the platform's features.	Messaging tickets should include sending emails to users upon updates (e.g to terms and policies)
5	**Password Reset Page Design**	Design the password reset flow	1. The form includes fields for both "New Password" and "Confirm Password." 2. The form includes a "Reset Password" button. 3. The form adheres to the platform's branding guidelines.	As a user, I want to set a new password after resetting it so that I can securely access my account.	Messaging tickets should include sending a mail for either OTP or a reset link
5	**Password Reset Page Design**	Design the password reset flow	@Prince @Temiloluwa Bajomo (Desktop, Tablet, Mobile)	1. The form includes fields for both "New Password" and "Confirm Password." 2. The form includes a "Reset Password" button. 3. The form adheres to the platform's branding guidelines. 4. If we are implementing an OTP, create a page for the user to input the OTP (Optional)	Settings ticket: As a user, I want to set a new password after resetting it so that I can securely access my account.
6	**Session Expiration**	Design the session expiration flow	@Olumide Adetifa - (Desktop, Tablet, Mobile).	1. When a user's session expires, the system must display an expiration page informing the user of the session expiration and providing clear instructions or messages about it. 2. The expiration page must include an option for the user to log in again and should maintain a consistent design with the rest of the application. 3. Upon selecting the log-in option, the user must be redirected to the login page.	Settings ticket: As a user, I want a seamless re-authentication process so that I can easily login.

	@Evelyn	
	@iamfavoureseosa	
	@ifa_dsgns	
	@Moses.ux	
	@Prince Chukwumezie	
	@web3deus	
	@bel	
	@ayo_sam	
	@Muklab	
	@Nifemi that guy	
	@Ellsa	
	@Temiloluwa Bajomo	
	@Emmanuel Nwoye	
	@Folowosele Funke	
	@Olumide Adetifa	
	@DesignBroke	