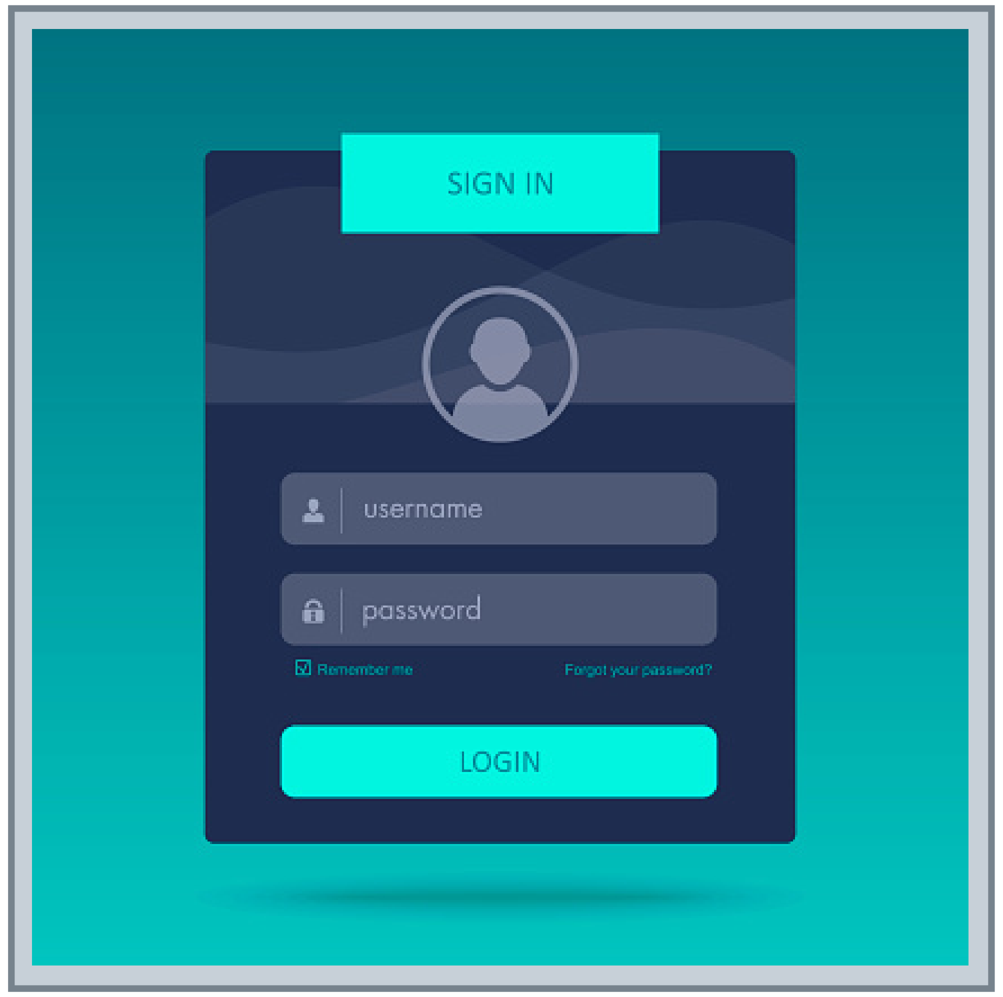
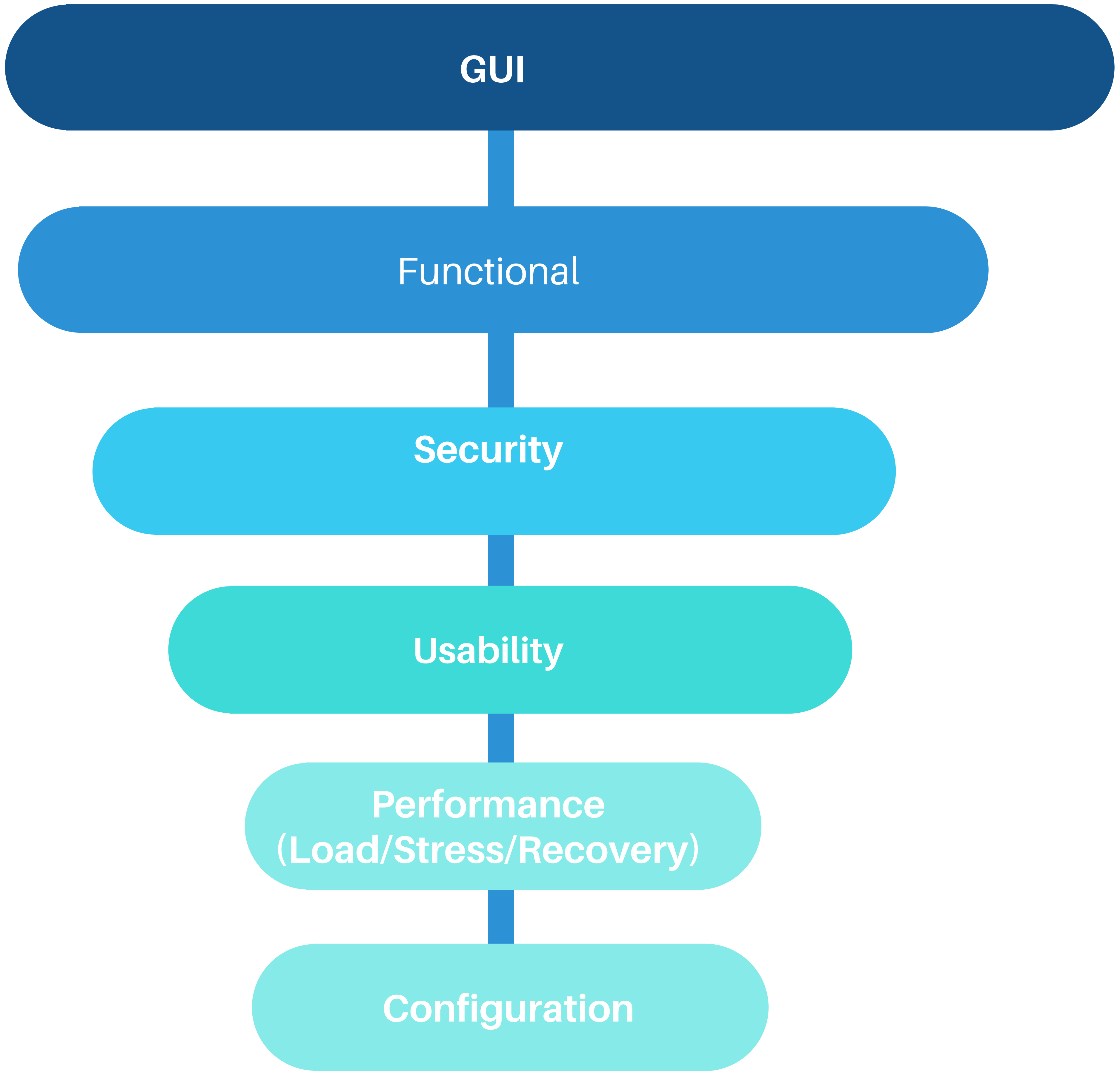
PLAN OF TESTING LOGIN FORM





GUI

Layout

- We check the presence of all elements; their size and color; location relative to each other;
- Comparison to layout made (overlaying a ready-made reference layout (psd file) on the application in the browser screen). There is a good tool Pixel Perfect for this;
- Measuring element dimensions. We can measure the dimensions of the element and compare them with the specification using Page Ruler;
- Correctness of fonts. We can check name, size, color of fonts and compare them with the specification using WhatFont;
- Interface colors - ColorZilla;
- Content - check for spelling and grammatical errors (SpellChecker);
- The appearance of the cursor - check presence and how the cursor appears at all in input fields, on clickable elements;
- HTML / CSS Standards - We can check it with W3C;
- Checking the presence and correctness of the favicon;
- Checking the presence and correctness of the title page;
- Scalability check (especially on the smartphones and tablets);
- Cross-browser compatibility(Chrome, Safari, IE, Firefox);
- Checking the Scroll;
- Browser extensions that can affect the appearance of the application (for example, AdBlock) - we try to enable and disable;
- Check content with disabled (WebDeveloper mode) images, flash, JavaScript.

FUNCTIONAL

Make sure that:

- The default cursor remains in the username text box when opening the login page;
- The user can navigate or access various controls by pressing the Tab key on the keyboard;
- Enter should work like Submit;
- You enter the password in a disguised form;
- The password can be copied or not;
- The user can log in by entering valid credentials and clicking the Login button.
- The user cannot log in with the wrong username and password;
- A validation message is displayed in case the user leaves the username or password field blank;
- A validation message is displayed in case the user exceeds the character limit for the username and password fields;
- Make sure the reset button works on the login page. Clicking on it clears the contents of the text box;
- All editing methods should be available (Insert, Delete, Backspace, Ctrl + C / V / X / Z, etc.). While copy-paste is allowed for the login field, but not desirable for the password field ;
- Make sure the "remember me" checkbox is checked on the login page;
- Make sure that closing the browser should not log out of the authenticated user.Running the application should only bring the user to the login state;
- Error messages: try to disable it in the browser settings;

Test Cases For a Login Page

- Enter the correct login and correct password. Expected: Logged in successfully. Log out. Clear cache and cookies (open / close browser). Leave both fields blank. Click on Login. Expected: alert;
- Leave the login field empty. Click on Login. Expected: alert;
- Leave the password field blank. Click on Login. Expected: alert;
- Enter the correct login and incorrect password. Expected: alert;
- Enter an incorrect login, but a correct password. Expected: alert;
- Enter an incorrect login and incorrect password. Expected: alert.;
- Enter login <script> alert (123) </script> and correct password. Expected: alert;
- Enter the SQL query into the login field ('or' a '=' a '; DROP TABLE user; SELECT ⋆ FROM blog WHERE code LIKE' a% ';) - the structure of the query depends on the DB;
- Enter html tags into the login field (<form action = " http://live.hh.ru" > <input type = "submit"> </form>);
- Enter a complex sequence of characters like "♣ ☺ ♂", "" '~! @ # $% ^ & ⋆ ()?>,. / \ <] [/ ⋆ <! - "", "$ {code } "; ->;
- Enter text consisting of only spaces in the login field;
- Enter the correct login, starting with several spaces, and the correct password in the login field. Expected: alert;
- Enter in the login field the correct login, followed by several spaces, and the correct password. Expected: alert;
- Enter the correct login and correct password. Click on the "Back" button in the browser. Expected: unclear;
- Enter the correct login. Specify a password using letters in a different case;
- Enter login using letters of DIFFERENT case. Enter the correct password.
- Register a user with the VasEA login. Expected: you can. Try to log in using only one case letters (vasea) in the login. Expected: you can;
- Register a user with the login petea / iZMaIL. Expected: you can. Try to log in using only one case letters in the password (petea / izmail). Expected: alert. Alert should point out the reason?;

- Check the limitation on the length of the login and password during registration? Enter qqweqweqweqweqweqweqweqweqweqweqweqweqwe / qqweqweqweqweqweqweqweqweqweqweqweqweqweqwe;
- Enter login / password Aa! @ # $% ^ & * () -_ + = `~ / \,.?> <| B / PaSSword! @ # $% ^ & * () -_ + =` ~ / \,.?> <| Are there any restrictions on valid characters? ;
- Enter username/password 'Иван', Is it possible to create with, for example, Cyrillic, if so, how does this form work then?;
- Enter the login ksjdksbdshdoueywfgjwevflwjeyfvowyecsydcvsldc (not present in the database), leave the password field empty. Expected: such user doesn't exist;
- Open the first browser. Log in as a valid user. Open a second browser. Log in with the same valid user. Expected: you can. Log out in the first browser. Expected: you can. Switch to the second browser. Do something that only a logged in user can do. Expected: you can;
- Open a browser. Enter valid data in the fields. Click on the Login button. Disable internet. Get “page unavailable”. Connect the internet back. Go on the website. Expected: not logged in;
- Log in with the correct username / password. Change password. Log in with a new password. Expected: the password has been changed, you can log in;
- Change password and log in under the old; remember password; Sign in; change password; log out; log back in with the old password; Expected: won't let you;
- Log in with the correct username / password. Rename account. Reload browser. Log in with your old username / password. Expected: won't let you. Log in with a new username / password. Expected: lets go;
- Log in with the correct username/password. Delete account. Reload browser. Log in with old username / password. Expected: won't let you;
- Enter the correct login and correct password. Copy the resulting url and paste it into another browser. Expected: It should not display the user's welcome page;
- How is a request to the server generated from this form (get / post)? How is the password transmitted - in the form of a hash or in plain text in the body of the POST? (If the data is transmitted in the address bar of the browser as "login = bla-bla & password = bla-bla"; apply all variants of incorrect data, including forbidden characters, and boundary values; - to transfer any parameter from the existing ones, for example. "Login = bla-bla & password = bla-bla & state = update")

Security

Make sure that:

- There is a limit on the total number of failed login attempts. To prevent the user from using a brute force mechanism to try all possible username and password combinations;
- If the credentials are incorrect, a message like “wrong username or password” should be displayed instead of a precise message indicating the wrong field. This is because the message like "wrong password" will help the hacker know that the username is correct and he just needs to try a different combination in the password field only;
- The length of the logon session timeout. Thus, once logged in, the user cannot be authenticated for all life;
- After logging in, pressing the Back button does not log the user out;
- SQL injection attacksdo not work on login page. The application should not be vulnerable to SQL injection attacks;
- Make sure the XSS vulnerability shouldn't work on the login page.

Usability

Make sure that:

- Whether the application meets the expectations of the end user - the form is easy to find on the product page,;
- The interface is logical;
- Compatibility with other software and hardware;
- Application speed;
- Information content (messages / required fields);

Performance (Load/Stress/Recovery)

- Simulate the number of users;
- Speed of execution of queries to the database;
- Change the Internet speed;
- Data / system recovery.

Configuration

- Get from the developers a list of software and hardware on which and with which our application should work and test in this environment;
- Thinking about what else the application interacts with (for example, social networks, mail, etc.);
- Check the different operating systems.