Form validation with Dialog box and Checkbox

Workshop #4

Flutter Developer Bootcamp

# **Purpose**

This workshop demonstrates how to implement basic form validation for Name and Email fields in a Flutter application using text controllers, regular expressions, and user interface components.

# **Problem**

In the given workshop, there will be two text fields: Name and Email. You need to make the form validation work when the name is not entered, and if an invalid email address is entered. Additionally, add a checkbox for terms and conditions, and include a dialog box displaying the entered values for the name field, email field, and agreement to terms condition.

# **How to Solve**

1. Checkout the workshop from Git Repo:

git clone -b <user-branch> <repo-URL>

2. Open the root folder inside VS Code

3. To build the app click the run option in the main method{}

4. Type an invalid Email in the field <textfield-name>

5. Click the Submit button, show a dialog box of field that have been entered in Name field, Email field and Agreed to terms condition.

6. Go To File: <specific-file-with-validation-method> à <method-name>, implement your validation logic. Make the validation work

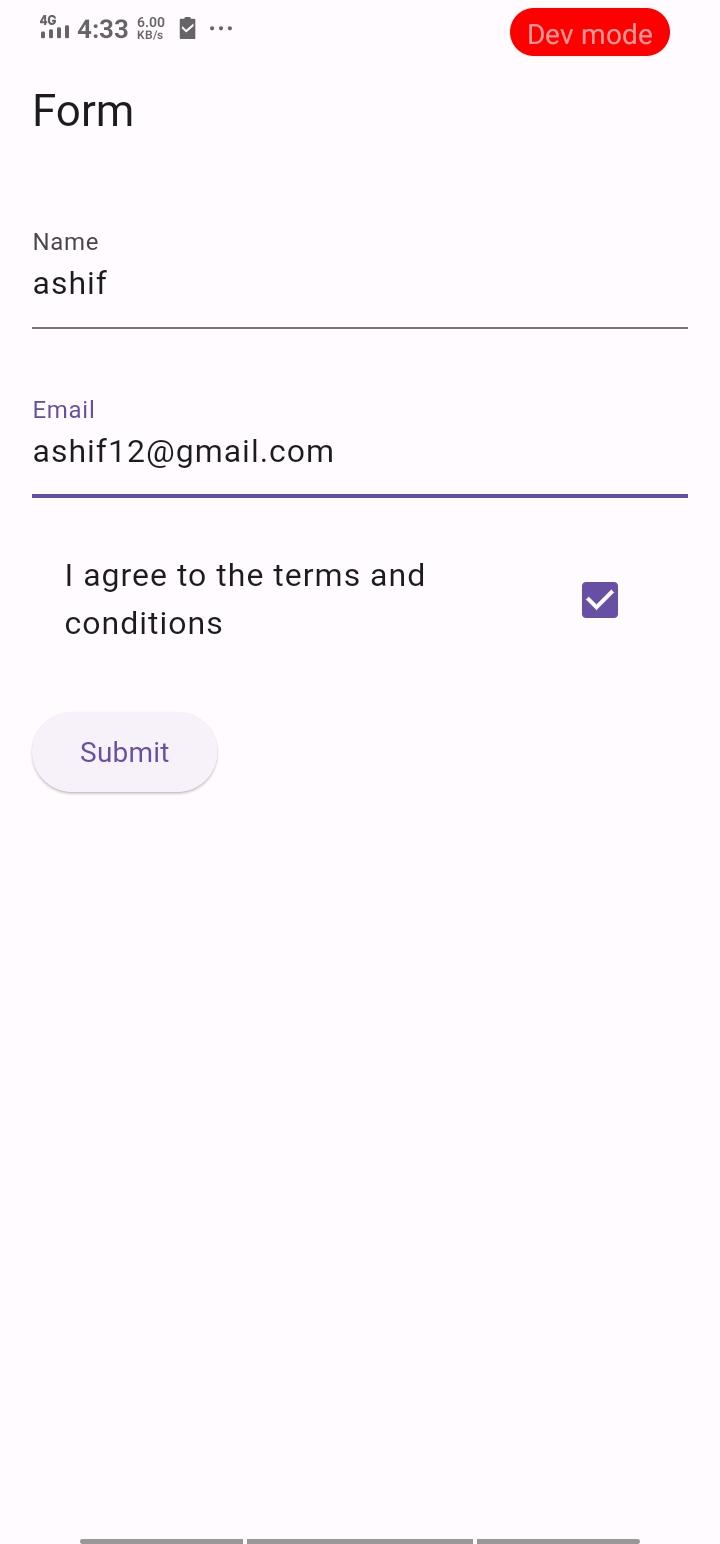
**You will Achieve**

When you complete this workshop you will learn the following:

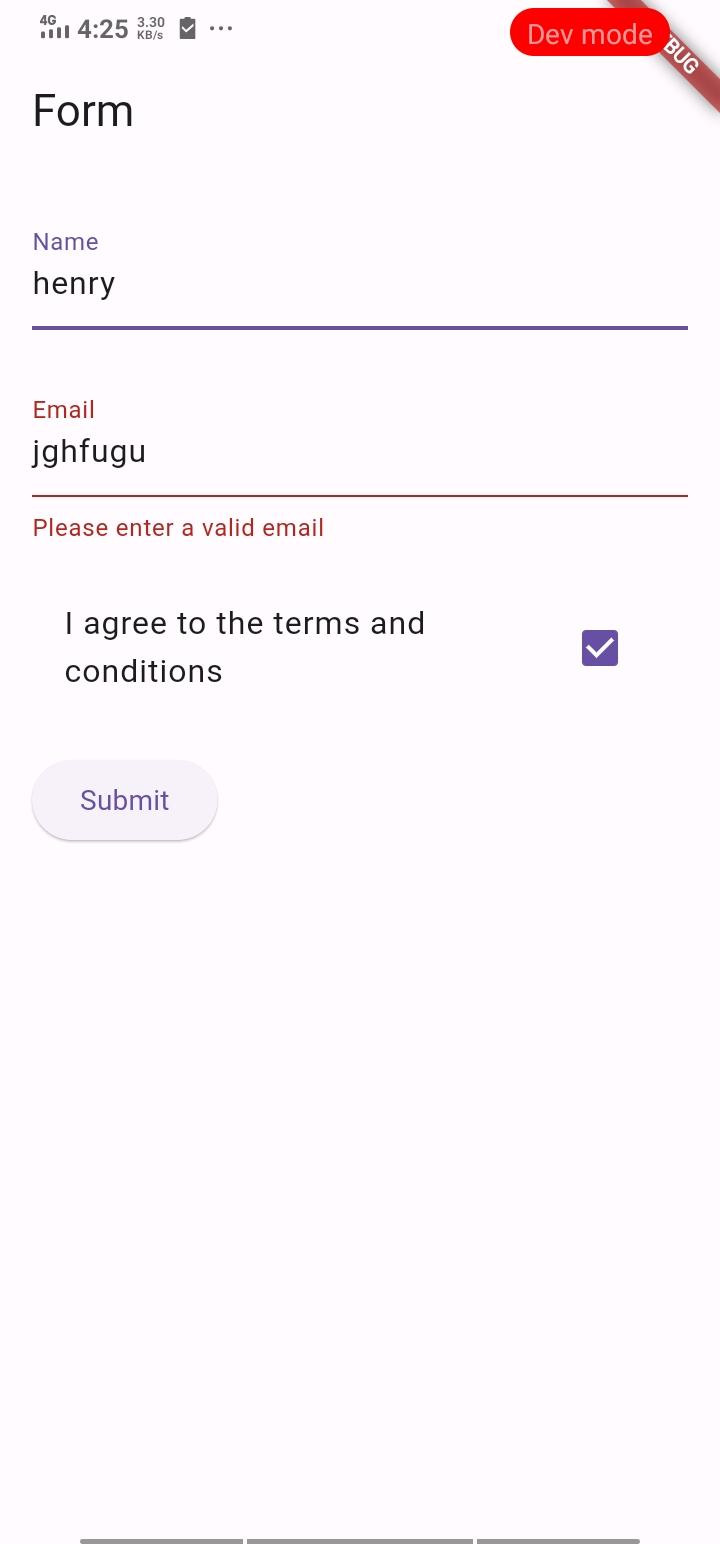
* **Regex Validation**: Learn how to use regular expressions to validate user input in text fields, ensuring that data entered by users meets specific criteria or formats.
* **Text**: Understand how to display textual content within your app's interface, including headings, paragraphs, labels, or other information.
* **TextFormField**: Implement text form fields for users to input text or data into your app, with built-in support for validation and formatting.
* **Label Text**: Incorporate label text alongside form fields to provide context or instructions to users, enhancing usability and clarity.
* **Decoration**: Explore techniques for applying decoration to widgets, such as text form fields or buttons, to enhance their visual appearance and style.
* **Buttons**: Integrate buttons into your app to trigger actions or navigate to different screens, improving user interactivity and navigation.
* **Checkboxes**: Add checkboxes to allow users to make selections or choices within your app, providing flexibility and customization options.
* **Dialog Box**: Utilize dialog boxes to display important messages, alerts, or prompts to users, improving user communication and interaction.
* **Padding**: Adjust spacing around widgets using the Padding widget to improve the layout and aesthetics of your app's user interface.
* **SizedBox**: Manage widget sizes and spacing within your app's layout using the SizedBox widget, providing flexibility and control over the layout's structure.
* **Adjusting Text Alignment**: Control the alignment of text elements within your app's interface to improve readability and visual consistency, enhancing the overall user experience.

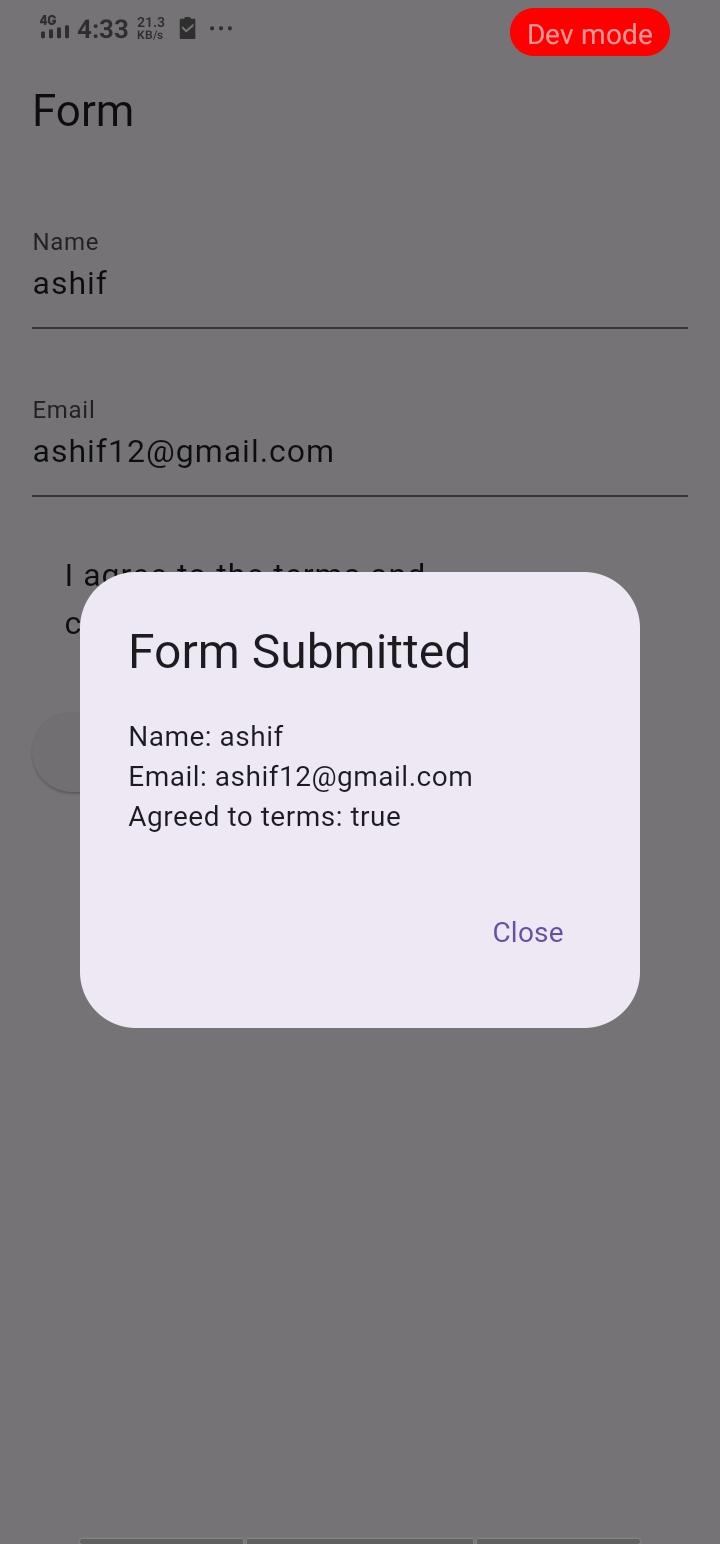
# **Screenshots**

## **Before implementation (without validation)**



## **After implementation (With validation)**





# **How to submit your workshop**

Push your project back to the same git branch using command:

<command name>

# **Happy Coding!**