Flutter 2 Focus

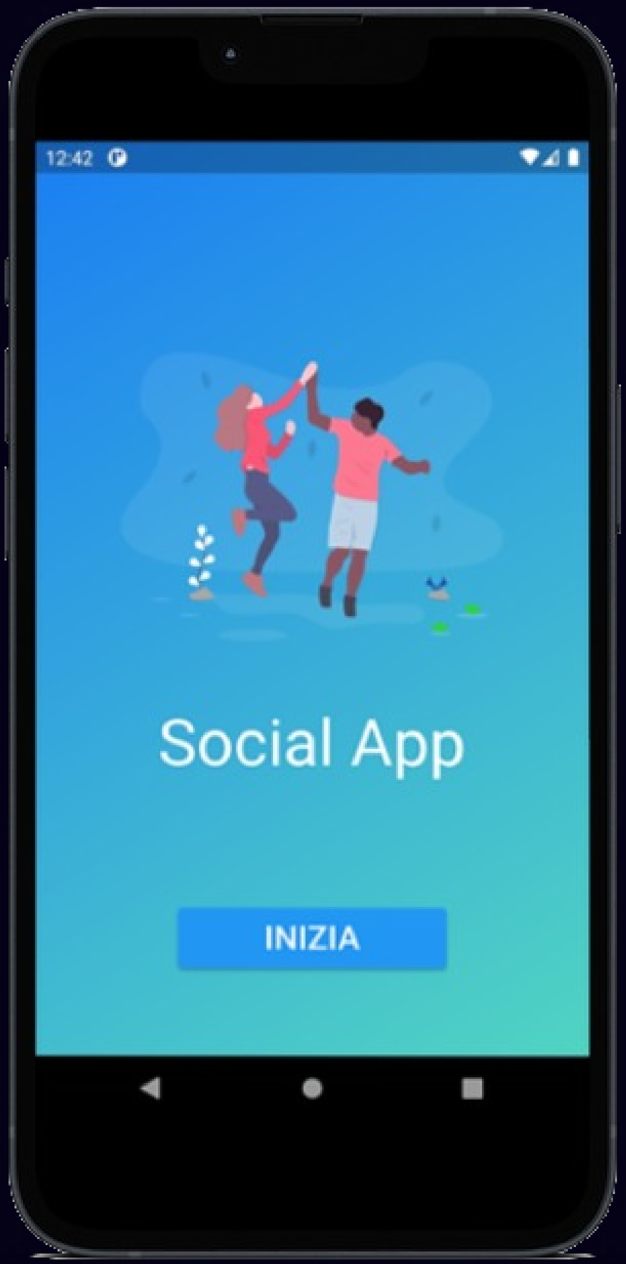
Introduzione

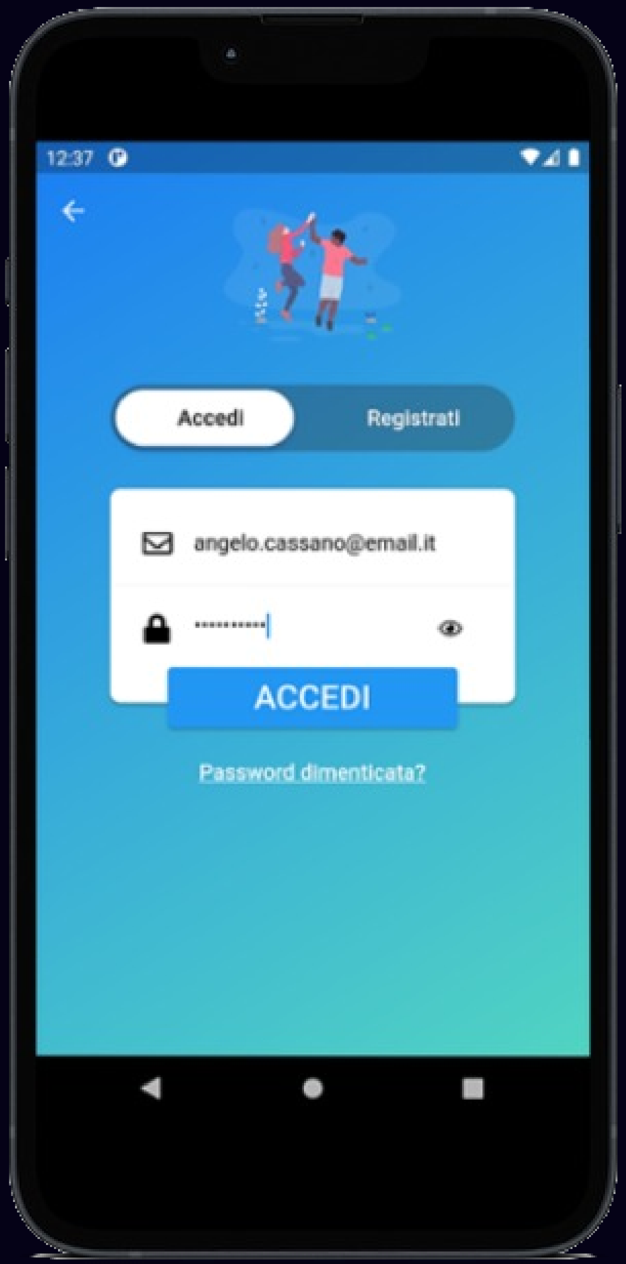
Fudeo

01

Flutter 2 Focus

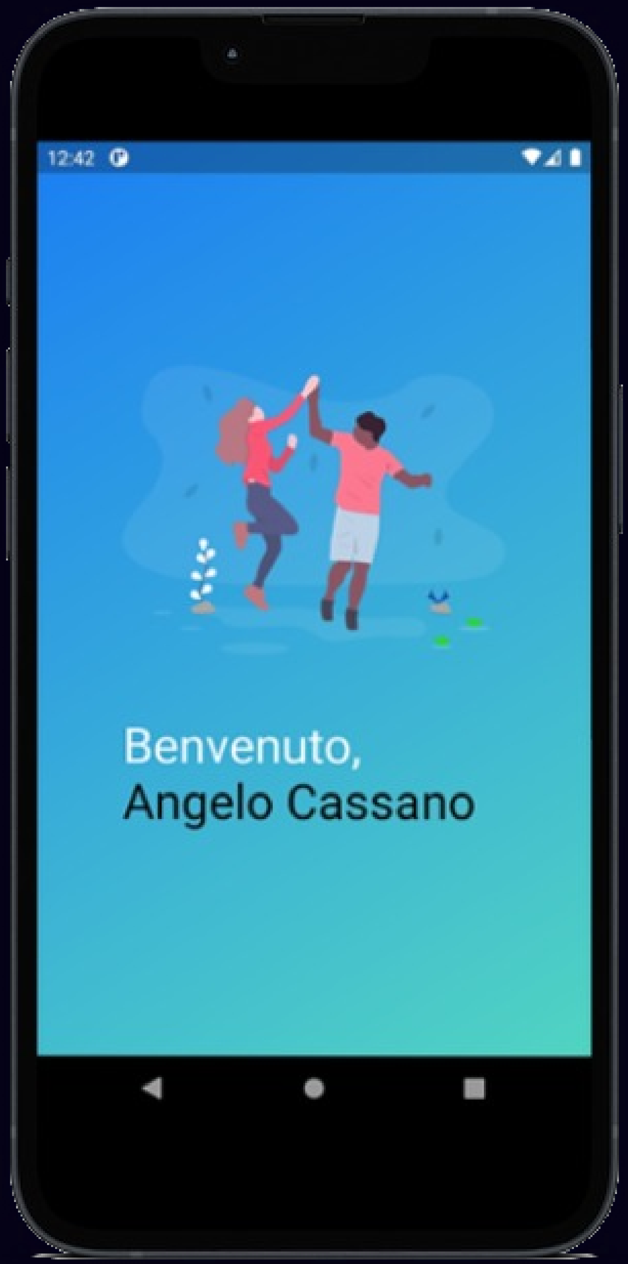
Form





angelo.cassano@email.it

••••••••••

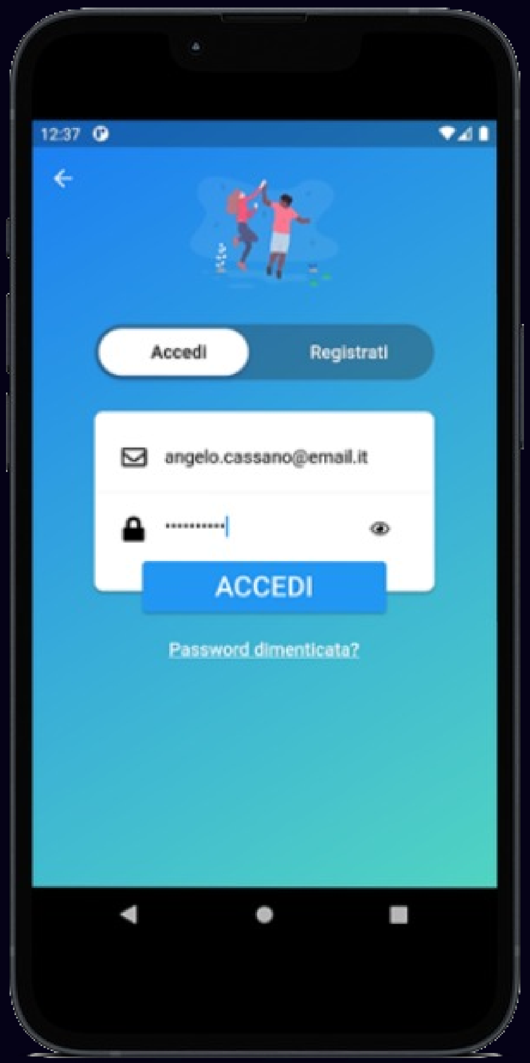


Benvenuto,
Angelo Cassano

02

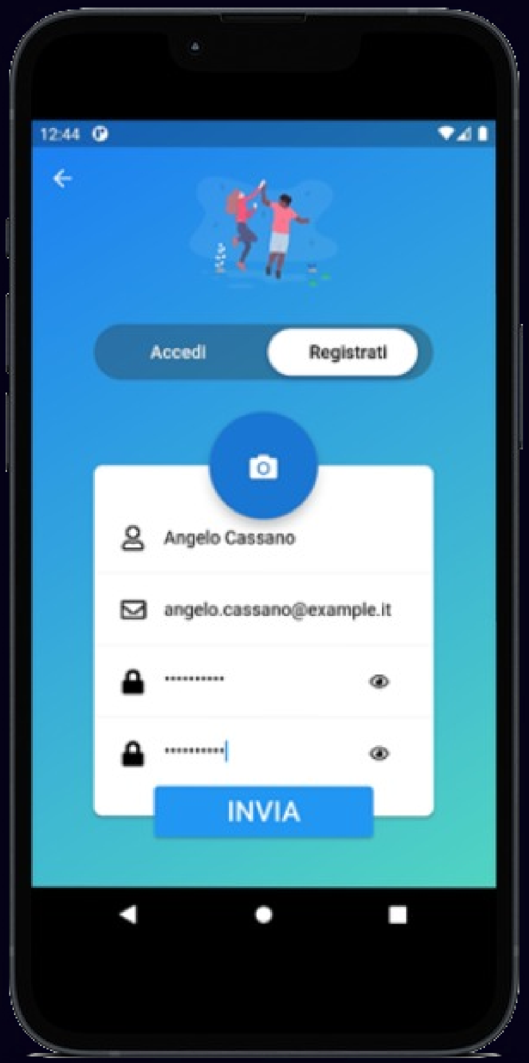
Flutter 2 Focus

Form



angelo.cassano@email.it

••••••••••



Angelo Cassano

angelo.cassano@example.it

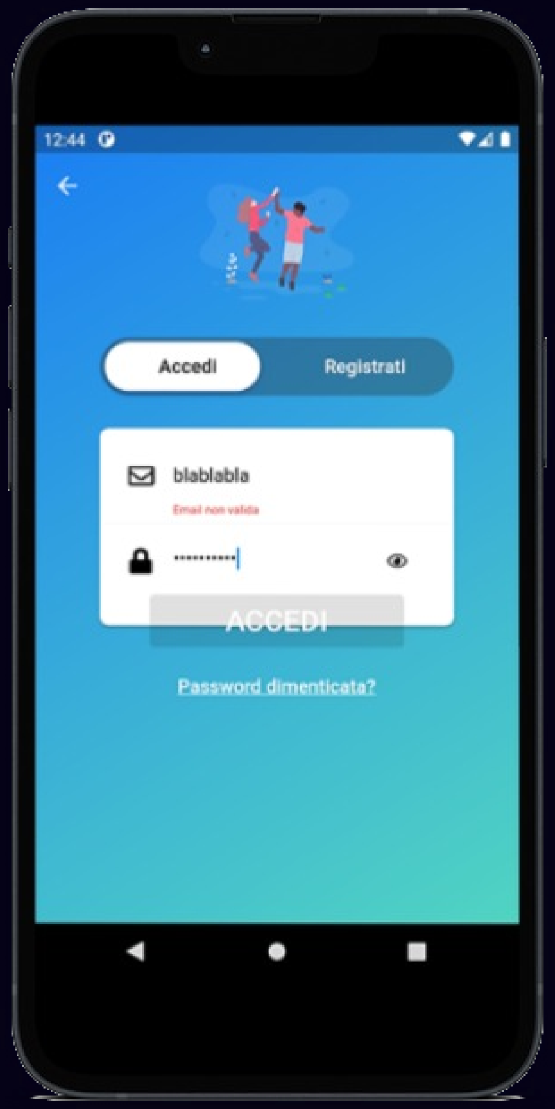
••••••••••

••••••••••

03

Flutter 2 Focus

Form



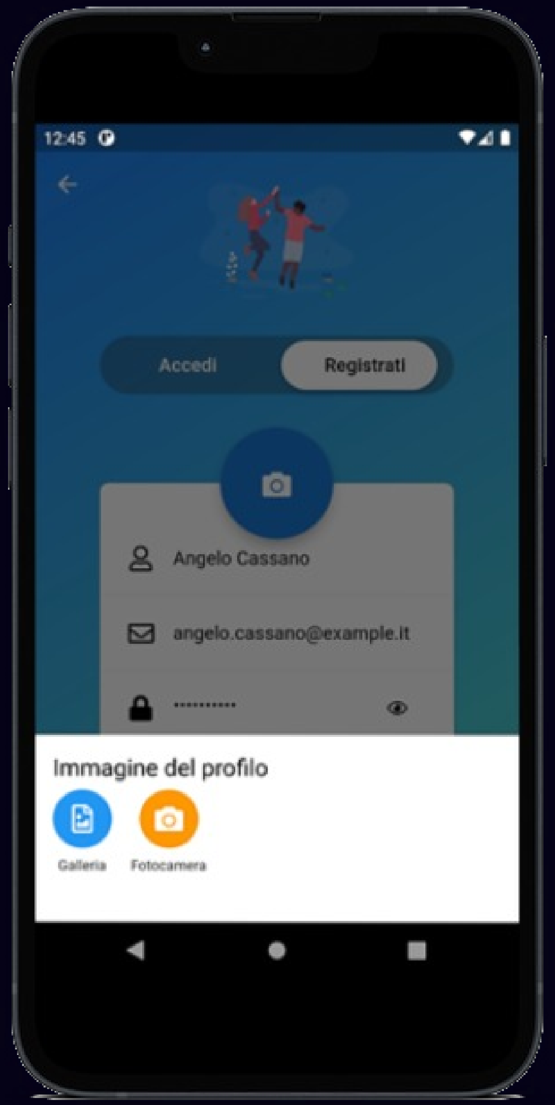
```
validator: (value) {
  if (value == null || value.length < 8) {
    return 'Password troppo corta';
  }
},
```

```
validator: (value) {
  if (!RegExp(
          r"^[a-zA-Z0-9.a-zA-Z0-9.!#$%&'*+-/=?^_`{|}~]+@[a-zA-Z0-9]+\.[a-zA-Z]+")
      .hasMatch(value ?? '')) {
    return 'Email non valida';
  }
},
```

04

Flutter 2 Focus

Form



Angelo Cassano

angelo.cassano@example.it

··········

Immagine del profilo

```
await Isolate.spawn(
  resizeImage,
  DecodeParam(
      source: sourceFile,
      destination: destinationFile,
      port: _receivePort.sendPort),
);


static void resizeImage(DecodeParam param) {
  final image = decodeImage(param.source.readAsBytesSync())!;
  final thumbnail = copyResize(image, width: 300);

  param.destination.writeAsBytesSync(encodePng(thumbnail));
  param.port.send(null);
}
```

05

Flutter 2 Focus

Form

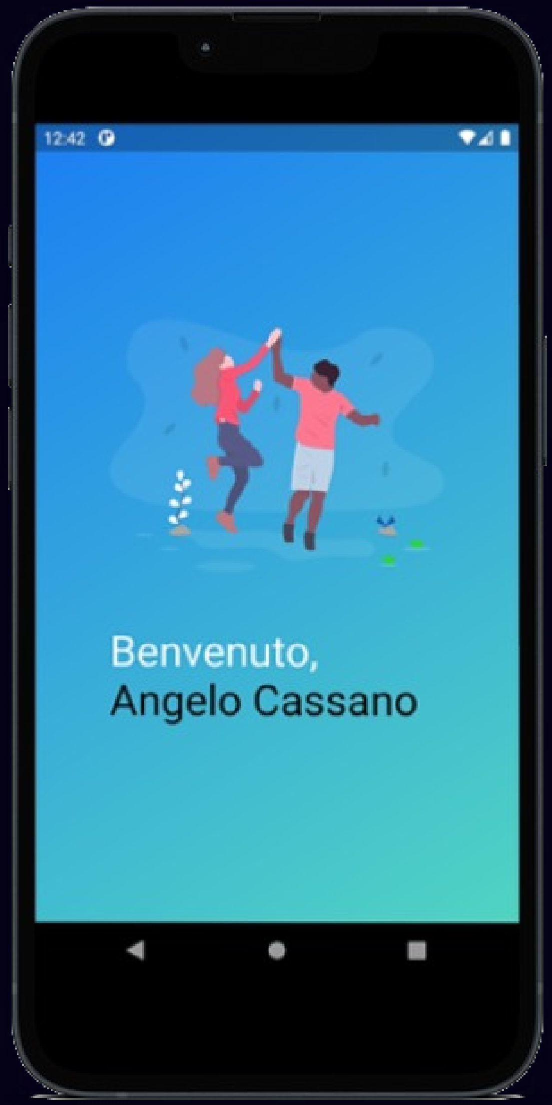
```
Future<User> signIn(SignInRequest request) async {
  final response = await http.post(
    Uri.parse('$baseUrl/login'),
    headers: {
      'Content-Type': 'application/json; charset=UTF-8',
    },
    body: jsonEncode(request.toJson),
  );

  if (response.statusCode == 200) {
    return User.fromJson(jsonDecode(response.body));
  }

  throw Exception('Failed to sign in');
}
```

```
final user = await authService.signIn(
  SignInRequest(
    email: email,
    password: password,
  ),
);

final encodedJson = user.toJson;
final json = jsonEncode(encodedJson);

await secureStorage.write(
  key: _currentUser,
  value: json,
);

return user;
```

06

Flutter 2 Focus

Form


12:42

Benvenuto,
Angelo Cassano

```
final encodedJson = user.toJson;
final json = jsonEncode(encodedJson);

await secureStorage.write(
  key: _currentUser,
  value: json,
);


Future<User?> get currentUser async {
 final json = await secureStorage.read(key: _currentUser);

 if (json != null) {
   final decodedJson = jsonDecode(json);
   return User.fromJson(decodedJson);
 }
}
```