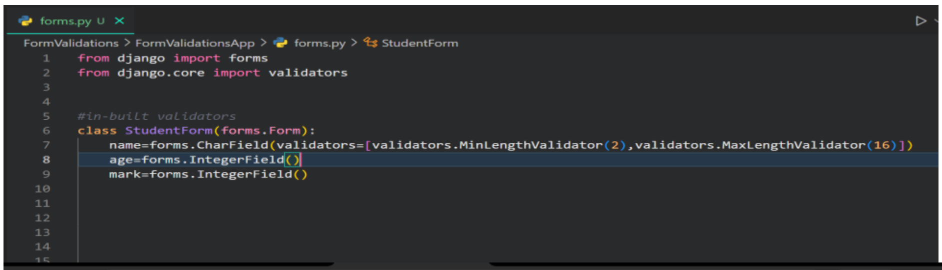
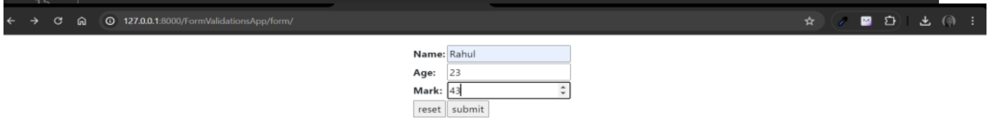
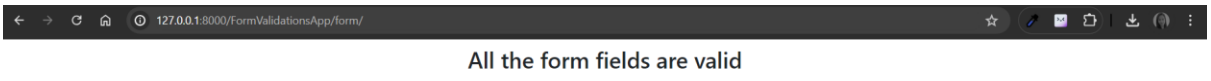

forms.py U
FormValidations > FormValidationsApp > forms.py > StudentForm

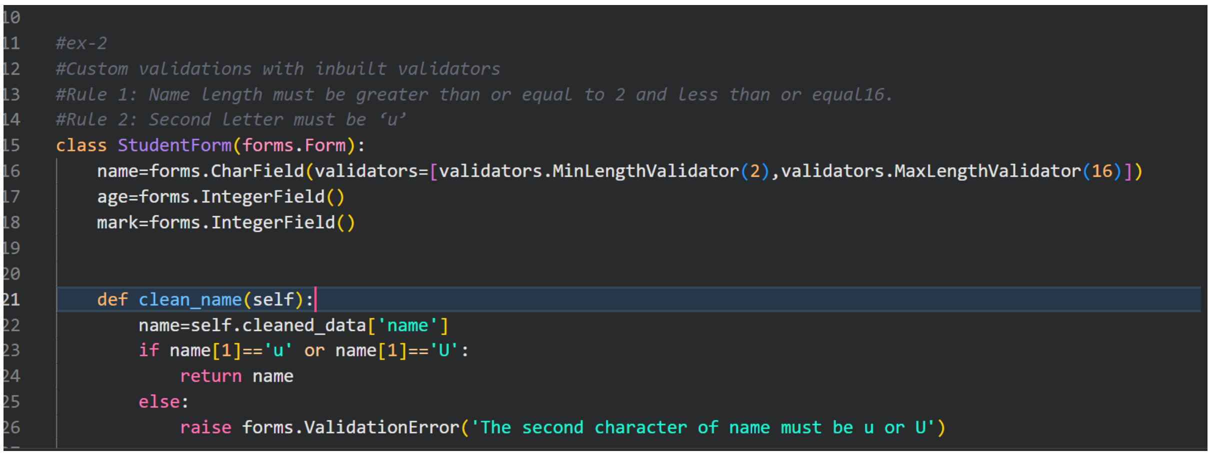
```python
from django import forms
from django.core import validators


#in-built validators
class StudentForm(forms.Form):
    name=forms.CharField(validators=[validators.MinLengthValidator(2),validators.MaxLengthValidator(16)])
    age=forms.IntegerField()
    mark=forms.IntegerField()
```



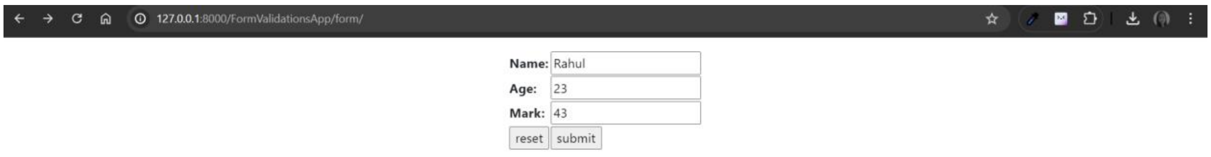
Name:	Rahul
Age:	23
Mark:	43



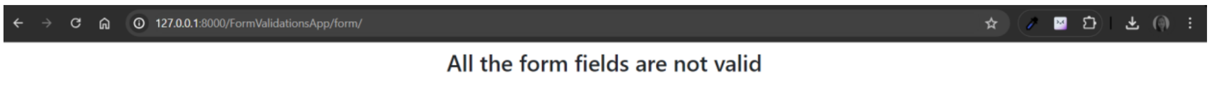
All the form fields are valid



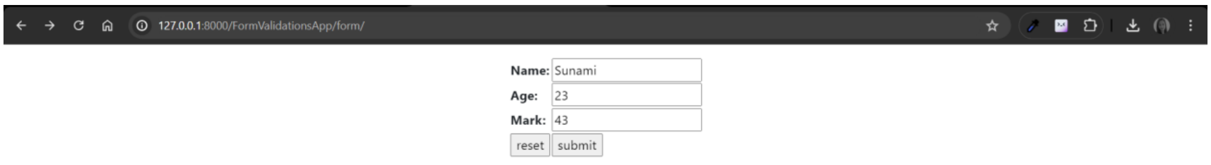
```python

#ex-2
#Custom validations with inbuilt validators
#Rule 1: Name length must be greater than or equal to 2 and less than or equal16.
#Rule 2: Second letter must be ‘u’
class StudentForm(forms.Form):
    name=forms.CharField(validators=[validators.MinLengthValidator(2),validators.MaxLengthValidator(16)])
    age=forms.IntegerField()
    mark=forms.IntegerField()


    def clean_name(self):
        name=self.cleaned_data['name']
        if name[1]=='u' or name[1]=='U':
            return name
        else:
            raise forms.ValidationError('The second character of name must be u or U')
```



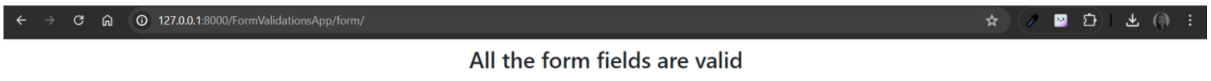
Name:	Rahul
Age:	23
Mark:	43



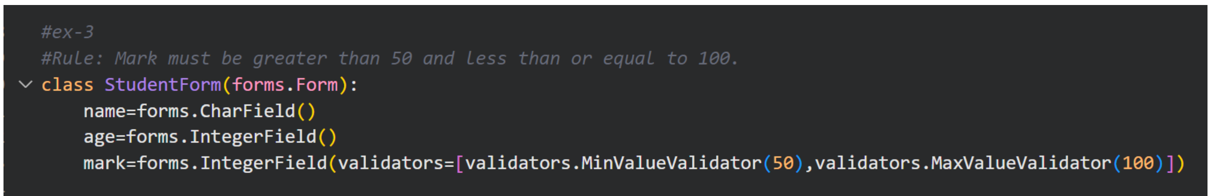
All the form fields are not valid

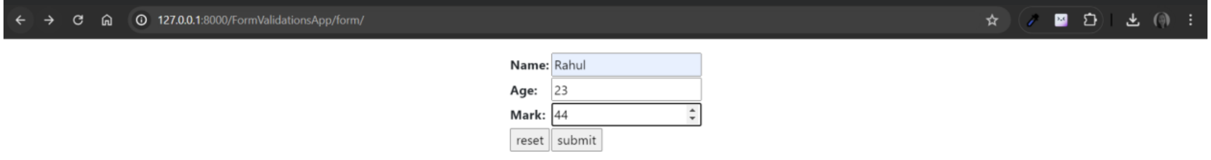

127.0.0.1:8000/FormValidationsApp/form/

Name: Sunami
Age: 23
Mark: 43

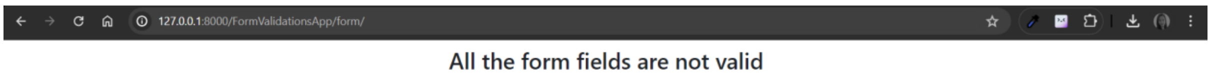

127.0.0.1:8000/FormValidationsApp/form/

All the form fields are valid

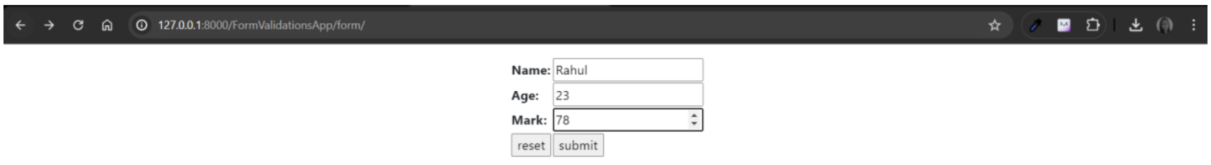

```
#ex-3
#Rule: Mark must be greater than 50 and less than or equal to 100.
class StudentForm(forms.Form):
    name=forms.CharField()
    age=forms.IntegerField()
    mark=forms.IntegerField(validators=[validators.MinValueValidator(50),validators.MaxValueValidator(100)])
```


127.0.0.1:8000/FormValidationsApp/form/

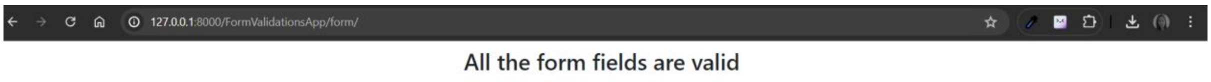
Name: Rahul
Age: 23
Mark: 44


127.0.0.1:8000/FormValidationsApp/form/

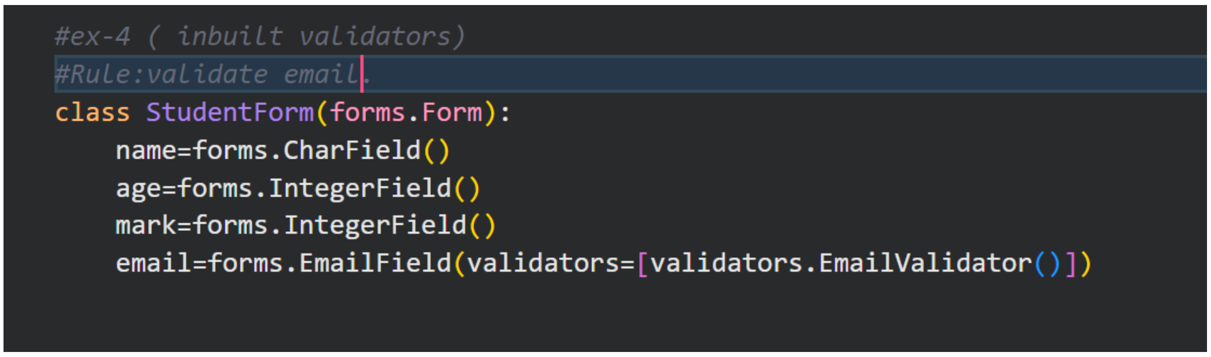
All the form fields are not valid

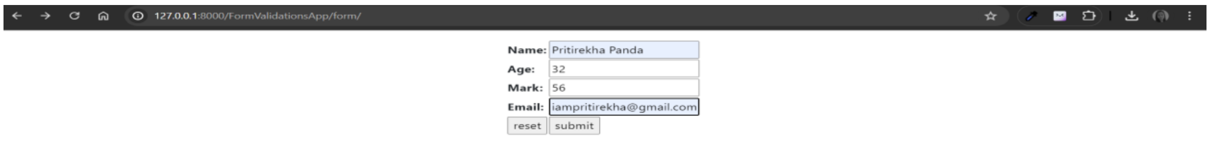

127.0.0.1:8000/FormValidationsApp/form/

Name: Rahul
Age: 23
Mark: 78


127.0.0.1:8000/FormValidationsApp/form/

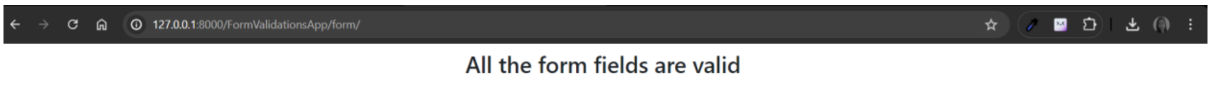
All the form fields are valid


```
#ex-4 ( inbuilt validators)
#Rule:validate email
class StudentForm(forms.Form):
    name=forms.CharField()
    age=forms.IntegerField()
    mark=forms.IntegerField()
    email=forms.EmailField(validators=[validators.EmailValidator()])
```

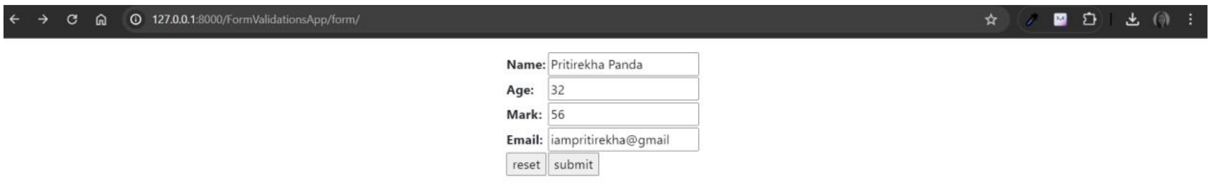

127.0.0.1:8000/FormValidationsApp/form/

Name:	Pritirekha Panda
Age:	32
Mark:	56
Email:	iampritirekha@gmail.com

reset submit

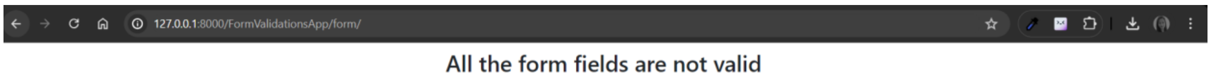

127.0.0.1:8000/FormValidationsApp/form/

All the form fields are valid

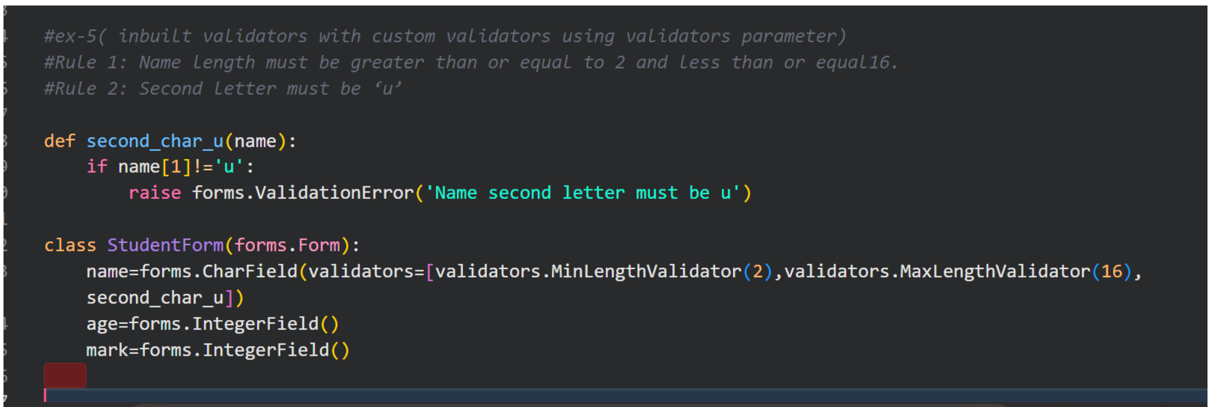

127.0.0.1:8000/FormValidationsApp/form/

Name:	Pritirekha Panda
Age:	32
Mark:	56
Email:	iampritirekha@gmail

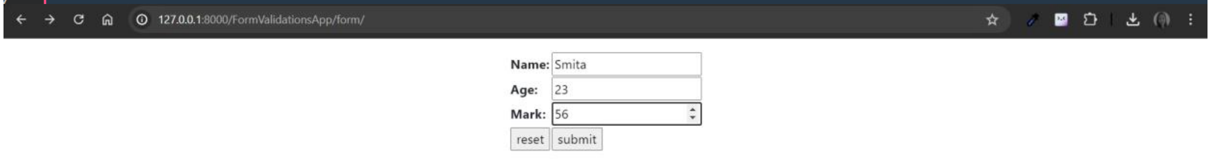
reset submit


127.0.0.1:8000/FormValidationsApp/form/

All the form fields are not valid

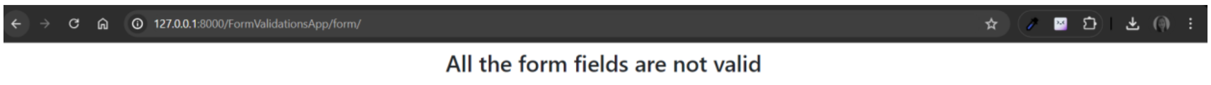


```
#ex-5( inbuilt validators with custom validators using validators parameter)
#Rule 1: Name length must be greater than or equal to 2 and less than or equal16.
#Rule 2: Second letter must be 'u'

def second_char_u(name):
    if name[1]!='u':
        raise forms.ValidationError('Name second letter must be u')

class StudentForm(forms.Form):
    name=forms.CharField(validators=[validators.MinLengthValidator(2),validators.MaxLengthValidator(16),
    second_char_u])
    age=forms.IntegerField()
    mark=forms.IntegerField()
```

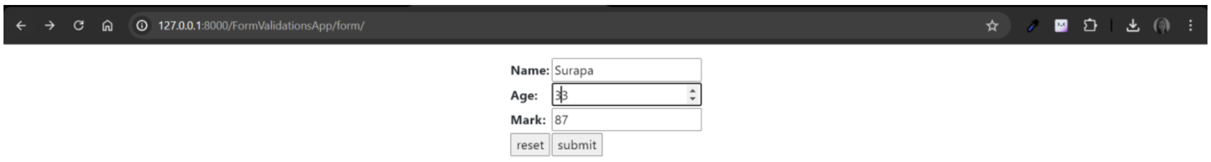

127.0.0.1:8000/FormValidationsApp/form/

Name:	Smita
Age:	23
Mark:	56

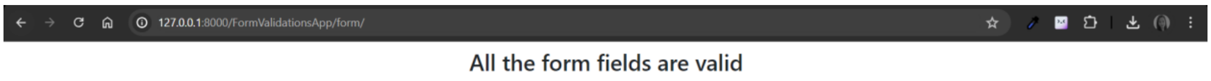
reset submit


127.0.0.1:8000/FormValidationsApp/form/

All the form fields are not valid



Name:	Surapa
Age:	33
Mark:	87



All the form fields are valid