from flask\_wtf import FlaskForm

from flask\_login import current\_user

from wtforms import StringField, PasswordField, SubmitField, BooleanField, SelectField, IntegerField

from wtforms.validators import DataRequired, Length, Email, EqualTo, ValidationError

from hired.models import User, Recruiter, Job, Job\_User

class RegistrationForm(FlaskForm):

username = StringField('Username', validators=[DataRequired(), Length(min=3, max=50)])

email = StringField('Email', validators=[DataRequired(), Email()])

password = PasswordField('Password', validators=[DataRequired(), Length(min=8)])

confirmPassword = PasswordField('Confirm Password', validators=[DataRequired(), EqualTo('password')])

remember = BooleanField('Remember Me')

submit = SubmitField('Sign Up')

def validate\_username(self, username):

user = User.query.filter\_by(username=username.data).first()

if user:

raise ValidationError('This username is taken. Please choose a different one.')

def validate\_email(self, email):

user = User.query.filter\_by(email=email.data).first()

if user:

raise ValidationError('This email is taken. Please choose a different one.')

class RecRegistrationForm(FlaskForm):

username = StringField('Username', validators=[DataRequired(), Length(min=3, max=50)])

email = StringField('Email', validators=[DataRequired(), Email()])

password = PasswordField('Password', validators=[DataRequired(), Length(min=8)])

confirmPassword = PasswordField('Confirm Password', validators=[DataRequired(), EqualTo('password')])

remember = BooleanField('Remember Me')

submit = SubmitField('Sign Up')

def validate\_username(self, username):

recruiter = Recruiter.query.filter\_by(username=username.data).first()

if recruiter:

raise ValidationError('This username is taken. Please choose a different one.')

def validate\_email(self, email):

recruiter = Recruiter.query.filter\_by(email=email.data).first()

if recruiter:

raise ValidationError('This email is taken. Please choose a different one.')

class LoginForm(FlaskForm):

email = StringField('Email', validators=[DataRequired(), Email()])

password = PasswordField('Password', validators=[DataRequired()])

remember = BooleanField('Remember Me')

submit = SubmitField('Login')

class RecLoginForm(FlaskForm):

email = StringField('Email', validators=[DataRequired(), Email()])

password = PasswordField('Password', validators=[DataRequired()])

remember = BooleanField('Remember Me')

submit = SubmitField('Login')

class UpdateAccountForm(FlaskForm):

username = StringField('Username', validators=[DataRequired(), Length(min=3, max=50)])

email = StringField('Email', validators=[DataRequired(), Email()])

degree\_type = StringField('Degree', validators=[DataRequired(), Length(min=2, max=7)])

experience = IntegerField('Experience(In Years)', validators=[DataRequired()])

major = StringField('Major', validators=[DataRequired(), Length(min=3, max=30)])

skill\_1 = StringField('Skill', validators=[DataRequired(), Length(min=2, max=10)])

skill\_1\_level = IntegerField('Skill Level (1-10)', validators=[DataRequired()])

dbms = StringField('DBMS you are most familiar with', validators=[DataRequired(), Length(min=2, max=10)])

submit = SubmitField('Update')

def validate\_username(self, username):

if username.data != current\_user.username:

user = User.query.filter\_by(username=username.data).first()

if user:

raise ValidationError('This username is taken. Please choose a different one.')

def validate\_email(self, email):

if email.data != current\_user.email:

user = User.query.filter\_by(email=email.data).first()

if user:

raise ValidationError('This email is taken. Please choose a different one.')

class RecUpdateAccountForm(FlaskForm):

username = StringField('Username', validators=[DataRequired(), Length(min=3, max=50)])

email = StringField('Email', validators=[DataRequired(), Email()])

location = StringField('Base Location', validators=[DataRequired(), Length(min=3, max=20)])

submit = SubmitField('Update')

def validate\_username(self, username):

if username.data != current\_user.username:

recruiter = Recruiter.query.filter\_by(username=username.data).first()

if recruiter:

raise ValidationError('This username is taken. Please choose a different one.')

def validate\_email(self, email):

if email.data != current\_user.email:

recruiter = Recruiter.query.filter\_by(email=email.data).first()

if recruiter:

raise ValidationError('This email is taken. Please choose a different one.')

class CandidateDetailsForm(FlaskForm):

degree\_type = StringField('Degree', validators=[DataRequired(), Length(min=2, max=7)])

experience = IntegerField('Experience(In Years)', validators=[DataRequired()])

major = StringField('Major', validators=[DataRequired(), Length(min=3, max=30)])

skill\_1 = StringField('Skill', validators=[DataRequired(), Length(min=2, max=10)])

skill\_1\_level = IntegerField('Skill Level (1-10)', validators=[DataRequired()])

dbms = StringField('DBMS you are most familiar with', validators=[DataRequired(), Length(min=2, max=10)])

submit = SubmitField('Submit Details')

class CreateJobForm(FlaskForm):

job\_id = IntegerField('Job Id', validators=[DataRequired()])

skill = StringField('Skill', validators=[DataRequired(), Length(min=2, max=10)])

location = StringField('Location', validators=[DataRequired(), Length(min=3, max=20)])

submit = SubmitField('Submit Details')

def validate\_email(self, job\_id):

if job\_id.data != Job.query.filter\_by(job\_id=job\_id):

raise ValidationError('This job id is taken. Please choose a different one.')

class SelectJobForm(FlaskForm):

job\_id = IntegerField('Enter the Job Id you want to apply for:', validators=[DataRequired()])

submit = SubmitField('Apply now')