



wie speichere ich Daten aus Flask-WTF-Formular nach dem Absenden in der Datenbank und aktualisiere den aktuell eingeloggten User, und dabei soll verhindert werden das eine email gespeichert wird, die bereits verwendet wurde und zusätzlich das ein eingeloggte Benutzer seine eigene aktuelle email weiterhin verwenden darf?

Du brauchst in deiner Profil-Route eine Kombination aus Formular-Validierung, E-Mail-Check in der Datenbank und einem Sonderfall für den eigenen Datensatz.

Beispielroute mit Flask-WTF, Flask-Login und SQLAlchemy

```
from flask import render_template, redirect, url_for, flash, request
from flask_login import login_required, current_user
from yourapp import app, db
from yourapp.models import User
from yourapp.forms import ProfileForm # dein Flask-WTF-Formular

@app.route("/profil", methods=["GET", "POST"])
@login_required
def profil():
    # Beim ersten Aufruf mit aktuellen Userdaten füllen
    form = ProfileForm(obj=current_user)

    if form.validate_on_submit(): # POST + alle Validatoren OK [web:14]
        # 1. Prüfen, ob die neue E-Mail schon von jemand ANDEREM benutzt wird
        existing = User.query.filter_by(email=form.email.data).first() # [web:13]
        if existing and existing.id != current_user.id:
            form.email.errors.append("Diese E-Mail wird bereits verwendet.")
            # Kein Commit, Formular mit Fehlermeldung erneut anzeigen
            return render_template("profil.html", form=form)

        # 2. current_user mit Formulardaten aktualisieren
        current_user.firstName = form.firstName.data
        current_user.name = form.name.data
        current_user.email = form.email.data
        current_user.phone = form.phone.data

        # 3. In DB speichern
        db.session.commit() # [web:8]

        flash("Profil erfolgreich aktualisiert.", "success")
```

```

# Optional: je nach Rolle weiterleiten
next_page = request.args.get("next")
if next_page:
    return redirect(next_page)
if current_user.role == "helfer":
    return redirect(url_for("helfer"))
return redirect(url_for("kunde"))

# GET oder ungültiges Formular
return render_template("profil.html", form=form)

```

Wichtige Punkte

- `ProfileForm(obj=current_user)` füllt beim GET die Felder mit den aktuellen Userdaten, beim POST überschreibt WTForms diese Werte automatisch mit den abgeschickten Daten.^[1]
- Der E-Mail-Check `existing = User.query.filter_by(email=form.email.data).first()` stellt sicher, dass keine E-Mail doppelt vorkommt.^[2]
- Die Bedingung `existing and existing.id != current_user.id` erlaubt dem eingeloggten Benutzer, seine eigene aktuelle E-Mail weiterzuverwenden, blockiert aber E-Mails anderer Nutzer.^[3]
- `form.validate_on_submit()` sorgt dafür, dass die übrigen WTForms-Validatoren (z.B. `Email()`, `DataRequired()`) laufen, bevor du in die Logik gehst.^[4]

Wenn du magst, kannst du mir deinen `User`-Model-Ausschnitt und deine `ProfileForm` posten, dann passe ich den Code 1:1 an dein Projekt an.



1. <https://stackoverflow.com/questions/24826978/flask-wtf-editing-a-model-using-wtform-form-constructor-pre-filling-the-form>
2. <https://stackoverflow.com/questions/46141867/how-to-filter-by-multiple-criteria-in-flask-sqlalchemy>
3. <https://www.perplexity.ai/search/5d1f6245-9e5f-489e-9d7f-7807d6cb0b89>
4. <https://www.digitalocean.com/community/tutorials/how-to-use-and-validate-web-forms-with-flask-wtf>
5. https://www.reddit.com/r/flask/comments/16jzptl/using_flaskwtf_forms_what_is_the_best_way_to/
6. <https://explore-flask.readthedocs.io/en/latest/forms.html>
7. <https://www.youtube.com/watch?v=oSG2AqetBJU>
8. <https://stackoverflow.com/questions/40535547/flask-sqlalchemy-filter-by-value-or-another>
9. https://www.reddit.com/r/flask/comments/13txfrp/cant_get_current_user_to_work_effectively/
10. https://www.reddit.com/r/flask/comments/1326agw/i_have_a_username_or_email_form_is_there_any_way/
11. <https://oneuptime.com/blog/post/2026-02-02-flask-wtf-validation/view>