# flask-posts

בפרק הקודם ראינו כיצד לשנות פרטי משתמש, ואיך לשמור קבצים(תמונות) דרך השרת ולהשתמש בהם בקוד.  
הפרק נעבוד על פוסטים- נראה איך ליצור פוסט חדש, ואיך לשמור אותו במסד.  
  
**יצירת פוסט חדש-**  
דבר ראשון כדי ליצור פוסט חדש, צריך שיהיה לנו דף עבור יצירת פוסטים.   
בשביל הדף נצטרך להוסיף טופס חדש שבו נכתוב את תוכן הפוסט.  
נלך ל-forms.py ונבנה מחלקה החדשה שנקראת PostForm שיורשת מ-FlaskForm. למחלקה אמורים להיות בסה"כ שלושה שדות: כותרת שאמורה להיות מטיפוס StringField , כפתור submit ותיבת טקסט לכתיבת תוכן הפוסט.  
תיבת הטקסט אמורה להיות מטיפוס ייחודי עבור כתיבה של הרבה תוכן, לכן נשתמש במחלקה TextAreaField של wtforms מה שאומר שנצטרך לייבא גם אותה. כמובן שנדרוש שלא יישארו שדות ריקים, כי לא נרצה פוסטים ריקים, לכן נוסיף validarots מתאימים:

from wtforms import ... ,TextAreaField

...

class PostForm(FlaskForm):

    title = StringField('Title', validators=[DataRequired()])

    content = TextAreaField('Content', validators=[DataRequired()])

    submit = SubmitField('Post')

ועכשיו נרצה להוסיף גם template מתאים. המבנה של ה-template יהיה דומה מאוד למבנה של שאר הדפים שמשתמשים בטפסים, כמו דף ההרשמה או account.   
נבנה דף חדש בתיקייה flask\_example/templates ונקרא לו create\_post.html או בכל שם אחר שתבחרו, ונעתיק מ- account.html את תוכנו ונשנה כדי שיתאים לטופס החדש:

{% extends "layout.html" %}

{% block content %}

<div class="content-section">

    <form method="POST" action="">

        {{ form.hidden\_tag() }}

        <fieldset class="form-group">

            <legend class="border-bottom mb-4">{{ legend }}</legend>

            <div class="form-group">

                {{ form.title.label(class="form-control-label") }}

                {% if form.title.errors %}

                    {{ form.title(class="form-control form-control-lg is-invalid") }}

                    <div class="invalid-feedback">

                        {% for error in form.title.errors %}

                            <span>{{ error }}</span>

                        {% endfor %}

                    </div>

                {% else %}

                    {{ form.title(class="form-control form-control-lg") }}

                {% endif %}

            </div>

            <div class="form-group">

                {{ form.content.label(class="form-control-label") }}

                {% if form.content.errors %}

                    {{ form.content(class="form-control form-control-lg is-invalid") }}

                    <div class="invalid-feedback">

                        {% for error in form.content.errors %}

                            <span>{{ error }}</span>

                        {% endfor %}

                    </div>

                {% else %}

                    {{ form.content(class="form-control form-control-lg") }}

                {% endif %}

            </div>

        </fieldset>

        <div class="form-group">

            {{ form.submit(class="btn btn-outline-info") }}

        </div>

    </form>

</div>

{% endblock content %}

נעבור ל-routes.py ונוסיף פונקציית ניתוב חדשה לדף create\_post.   
הפונקציה אמורה לקבל מתודות post ו-get , לפעול רק אם יש משתמש מחובר כמובן, והיא אמורה לשלוח גם אובייקט מטיפוס טופס פוסט (PostForm).  
צריך לבדוק אם התקבל פוסט תקין, ואם כן צריך לשמור את התוכן שלו במסד הנתונים, לנתב לדף הבית ולשלוח הודעת flash שהפוסט החדש התקבל ונשמר:

from flask\_example.forms import RegistrationForm,...,PostForm

...

@app.route("/post/new", methods=['GET', 'POST'])

@login\_required

def new\_post():

    form = PostForm()

    if form.validate\_on\_submit():

        post = Post(title=form.title.data, content=form.content.data, author=current\_user)

        db.session.add(post)

        db.session.commit()

        flash('Your post has been created!', 'success')

        return redirect(url\_for('home'))

    return render\_template('create\_post.html', title='New Post',

                           form=form, legend='New Post')

וכמובן יש להוסיף אותו ל-layout.html (בערך שורה 34):

...  
<!-- Navbar Right Side -->

            <div class="navbar-nav">

              {% if current\_user.is\_authenticated %}

              <a class="nav-item nav-link" href="{{ url\_for('new\_post') }}">New Post</a>

              <a class="nav-item nav-link" href="{{ url\_for('account') }}">Account</a>

              <a class="nav-item nav-link" href="{{ url\_for('logout') }}">Logout</a>

            {% else %}

              <a class="nav-item nav-link" href="{{ url\_for('login') }}">Login</a>

...

לסיום החלק הזה נרצה שיראו את הפוסט החדש באתר ולא את ההודעה הדיפולטיבית שמוצגת בינתיים.   
נעלה לפונקציית הניתוב home() ,ונשנה אותה כך שתשלח רק הודעות מתוך ההטבלה של posts , ולא את מה שהיא שולחת כרגע :

@app.route("/")

@app.route("/home")

def home():

    posts = Post.query.all()

    return render\_template('home.html', posts=posts)

נעבור ל-home.html:  
אנחנו רוצים שעבור כל post ברשימת הפוסטים תוצג כותרת הפוסט, המשתמש שכתב את הפוסט, התמונה שלו, תאריך ותוכן הפוסט:

{% extends "layout.html" %}

{% block content %}

    {% for post in posts %}

        <article class="media content-section">

          <img class="rounded-circle article-img" src="{{ url\_for('static', filename='profile\_pics/' + post.author.profile\_img) }}">

          <div class="media-body">

            <div class="article-metadata">

              <a class="mr-2" href="#">{{ post.author.username }}</a>

              <small class="text-muted">{{ post.date\_posted.strftime('%Y-%m-%d') }}</small>

            </div>

            <h2><a class="article-title" href="#">{{ post.title }}</a></h2>

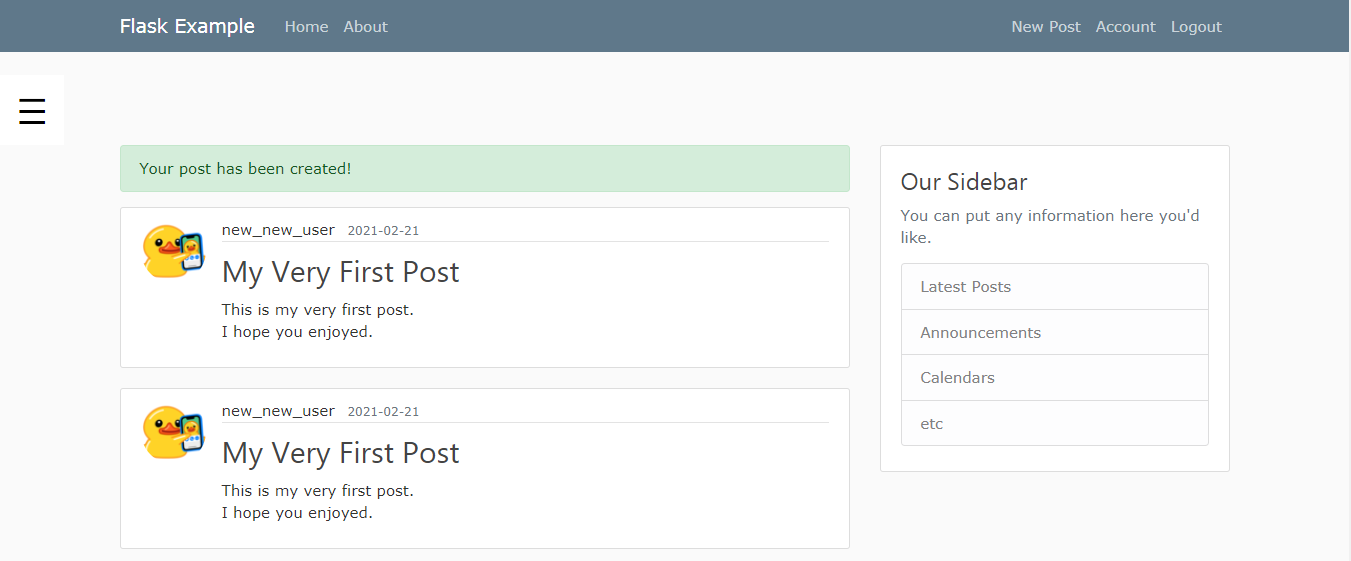
            <p class="article-content">{{ post.content }}</p>

          </div>

        </article>

    {% endfor %}

{% endblock content %}

נשמור ונריץ.   
אם עשינו הכל כמו שצריך אנחנו אמורים לראות את האתר ריק מתוכן , כי לא נכתב עדיין שום פוסט, ואם נתחבר עם משתמש אנחנו אמורים גם לראות את האפשרות לעבור לעמוד new post . אם ניצור פוסט חדש אנחנו אמורים לעבור אוטומטית לעמוד הבית ולראות את הפוסט שנכתב.  
  
  


ברמת העיקרון סיימנו את החלק של flask בקורס , אבל יש עוד הרבה מאוד מה להרחיב בנושא.   
נזכיר שאת כל חלק זה של הקורס לקחנו מסדרת שיעורים של ערוץ היוטיוב של Corey Schafer. סדרת השיעורים של הערוץ ממשיכה הרבה מעבר למה שאנחנו הצגנו: שליחת מייל למשתמש ושינוי סיסמא דרך המייל, pagination של פוסטים (שליטה בהצגת ההודעות), וכמובן העלאת הפרויקט לשרת אינטרנטי במקום שימוש ב-localhost. בקורס ניסינו להקנות את הכלים לשימוש בסיסי ב-flask ברמה שתוכל לתת לכם את העצמאות להרים פרויקטים מושקעים מהתכנים שלמדנו לכל אורך הקורס. כמובן שלצערנו לא הספקנו לעבור אפילו על רוב החומר בשל מגבלת הזמן, לכן נמליץ בחום להמשיך את סדרת השיעורים של קורי, ולחקור עוד בעצמכם בנושא.