TDP003 Projekt: Egna datormiljön

Installationsmanual

Författare
Maximilian Bragazzi Ihrén maxbr431@student.liu.se
Markus Lewin, marle943@student.liu.se

Höstterminen 2014
Version 1.1

1. Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
1.0	Skriven inför inlämning	141008
1.1	Lagt till information om loggning, fixat diverse kommentarer	141016

1. Förberedelser

Här går vi igenom hur man installerar Flask och Jinja2, vilka behövs för att du ska kunna starta din hemsida.

Se till att du har alla filer nerladdade, genom att ladda ner och packa upp följade ”.zip”-fil i valfri mapp:

http://www-und.ida.liu.se/~maxbr431/Flask_portfolio/flask_portfolio_maxbr431_marle943.zip

Denna ”.zip”-fil innehåller alla delar till hemsidan.

1.1. Installera Flask och Jinja2 på din dator

Flask är programvaran du kör när du ska starta din server. Flask är ett ramverk som håller reda på allt din kod, och håller servern uppe.

Detta steg är frivilligt, men rekommenderat då det kan behövas om du vill prova att titta på din hemsida innan du uppdaterar den på din server någon gång. Om du ser till att allt fungerar som det ska innan du lägger upp det på din server, så kan det bli jobbigt att uppdatera din sida till en fungerade version. Det är så klart bättre att ha en äldre, men fungerande, version än en nyare men ej fungerande version av din hemsida på din server.

Det första du gör är att öppna ditt terminalfönster, och skriver in:

```
sudo apt-get update
```

Det kommandot uppdaterar alla dina programvaror (det är alltid bra att göra, och det rekommenderas att du gör det här med jämna mellanrum). Om det kommer upp en prompt där den kräver ett lösenord, så skriver du bara in lösenordet som du har som inloggning till datorn. Om denna prompt kommer upp igen, skriv bara in lösenordet igen.

Sedan så skriver du in:

```
sudo apt-get install python-setuptools
```

python-setuptools är ett verktyg som hjälper dig att installera och underhålla Pythonpaket (så som Flask). Det enda vi kommer använda verktyget till i denna manual är att installera Flask.

Sedan går du in på följande hemsida:

https://packaging.python.org/en/latest/installing.html#setup-for-installing-distribution-packages

Ladda ner filen ”get-pip.py” som finns länkad där på hemsidan. Spara den till valfri mapp.

Navigera till mappen du sparade filen i med hjälp av kommandot ”cd”. Om du till exempel sparade filen på din Desktop skulle det se ut så här:

cd Desktop

Om det kommandot inte fungerar, prova att skriva bara ”cd” först, och sen skriva in det igen. Om du bara skriver ”cd” kommer du nämligen till din hemmapp på datorn, och Desktop ligger i denna mapp.

När du väl är i mappen du sparat ”get-pip.py” i, så skriver du:

sudo python get-pip.py

Det kommandot installerar då ”pip”-funktionen, vilket behövs för att installera Flask.

Skriv sedan:

sudo pip3.4 install flask

Vilket då till slut installerar Flask till din dator eller server. Installationen för Jinja2 är medföljande i installationen av Flask

1.1.1. Testa att Flask fungerar på din dator

Navigera till mappen där du laddade ner alla filer om hemsidan i terminalen. I den mappen ska det finnas en fil vid namn ”flask_check.py”. Kör denna fil med Python genom att skriva följande in i terminalen:

python flask_check.py

Det borde då komma upp en prompt i terminalen som säger:

* Running on http://127.0.0.1:5000/

Öppna då din browser och skriv in ”localhost:5000” in i ditt URL-fält. Det borde då visa en sida där det står:

Flask fungerar! Den kan även ta svenska bokstäver: å ä ö

Om det inte står som ovan, gå igenom installationen igen. Annars är det bara att trycka Ctrl + C i terminalfönstret för att stänga ner servern.

1.2. Installera Flask och Jinja2 på server

Börja med att öppna ditt terminalfönster och skriv in:

ssh (adressen till din server)

Det kan till exempel se ut som:

ssh server1.google.se

Du kommer då att promptas med ett krav på lösenord till servern. Skriv in det. Din dator kommer nu att komma ihåg ditt lösenord och inte prompta dig igen för ens du öppnar ett nytt terminalfönster.

När du nu är inne på servern så skriver du först:

sudo apt-get update

Det kommandot uppdaterar alla dina programvaror på servern, vilket alltid är bra att hålla på de senaste versionerna.

Skriv sedan:

logout

Nu är du tillbaka i din egen dator. Ladda då ner filen ”get-pip.py” (se punkt 1.1 för en länk och förklaring). Navigera dig till mappen där du sparade filen (se punkt 1.1 för instruktioner hur).

Skriv sedan in:

scp get-pip.py (adressen till din server)

Det kan till exempel se ut som:

scp get-pip.py server1.google.se

scp är ett kommando som kopierar filer över en internetanslutning. Du kopierar alltså filen get-pip.py till din server.

Logga nu in på servern igen:

ssh (adressen till din server)

När du nu är inne i servern igen skriver du:

sudo apt-get install python-setuptools

python-setuptools är ett verktyg som hjälper dig att installera och underhålla Pythonpaket (så som Flask). Det enda vi kommer använda verktyget till i denna manual är att installera Flask.

Installera sedan filen ”get-pip.py”, och efter det installera flask enligt instruktioner i punkt 1.1.

Vilket då till slut installerar Flask till din dator eller server. Installationen för Jinja2 är medföljande i installationen av Flask

1.2.1. Testa att Flask fungerar på din server

Se till att du är på din egen dator, och inte inne på servern. Om du är inne på servern, logga ut med hjälp av ”logout”-kommandot.

Navigera dig sedan till mappen där du laddade ner alla filer för servern via terminalen. I den mappen ska det finnas en fil vid namn ”check_flask.py”. Skicka denna fil till din server med hjälp av ”scp”-kommandot. Till exempel:

scp check_flask.py server1.google.se

Logga sedan in på din server med hjälp av ”ssh”-kommandot, så som ovan. Sedan så navigerar du till mappen där din ”check_flask.py”-fil lades, och kontrollera sedan enligt instruktionerna under punkt 1.1.1.

När allt fungerar och du har stängt ner din server kan du logga ut genom att skriva:

logout

2. Lägga in dina projekt i systemet

I mappen du laddade ner som innehöll hemsidan borde du hitta en fil vid namn ”data.json”. Öppna

denna fil i valfri textläsare (t.ex. Microsoft Word, LibreOffice Writer, eller Notepad).

Där i filen ser du ett exempelprojekt i det formatet som projekten ska finnas. Det ska alltså se ut så som följande (fast med mer data):

Allt som är obligatorisk data är project_name (projektets namn) och project_no (ett unikt heltal som ID för detta projekt. Det är rekommenderat att bara börja med 1 och sedan räkna uppåt). Det är dock rekommenderat att skriva in all data som står med i exempelprojektet.

Ett projekt är då alltså alla saker som finns inom en '{' och en '}'. Alla projekt ska omfamnas av en '[' och en ']', och det ska vara ett kommatecken efter varje '}' förutom den sista. Detta är för att en JSON-fil är kodad så att allt inom '[' och ']' är en lista, och varje element i listan är separerat med ett kommatecken. Liknande är det mellan '{' och '}', då alla utom det sista elementet ska avslutas med ett kommatecken.

Ett undantag är då elementet 'techniques_used', som är en lista i sig. Se till att alla element i listan slutar med ett kommatecken (förutom den sista, igen), men de ska inte omfamnas av '{' och '}' (Se bild).

2.1. Beskrivningar av alla element

- project_name – Projektets namn (t.ex ”Python kalkylator” eller ”Träbord”).
- project_no – Ett unikt heltals-ID för varje projekt.
- start_date – Datumet du startade ditt projekt på.
- end_date – Datumet du avslutade ditt projekt.
- course_id – ID för en eventuell kurs du gjort projektet i.
- course_name – Namnet på den eventuella kursen.

- academic_credits – Hur många poäng denna eventuella kurs var värd.
- techniques_used – En lista med tekniker som du använt under ditt projekt (t.ex. ”Python” eller ”Svarvning”).
- short_description – En kort beskrivning av ditt projekt.
- long_description – En lång beskrivning av ditt projekt.
- small_image – Bilden som visas överallt förutom på projekts egna sida. Skriv in samma som i big_image för att få samma bild. Glöm ej att skriva med filändelsen (t.ex ”.png”).
- big_image – Bilden som visas på projektets egna sida. Skriv in samma som i small_image för att få samma bild. Glöm ej att skriva med filändelsen (t.ex ”.png”).
- group_size – Hur många det var i gruppen som gjorde programmet. Om du var ensam, skriv 1
- external_link – En länk till projektets sida. Är formatterat som: ”/project/” och sedan projektets ID (t.ex. ”/project/1”)

2.2. Lägg till dina bilder

När du väl fyllt i allt om ditt projekt, kopiera de bilder du har angett som small_image och big_image till mappen ”images” som ligger i mappen ”static” i den mappen som du laddade ner med alla filer om hemsidan. Se till att namnet på bilden matchar bildens faktiska namn.

3. Lägga upp hemsidan på din server

Börja med att navigera till mappen där du laddade ner hemsidans filer med hjälp av ”cd”-kommandot. Gå till mappen ovanför den som filerna ligger i (t.ex. ”Hämtade Filer”). Du kan skriva:

```
cd ..
```

Det kommandot flyttar dig nämligen upp ett steg bland mapparna.

Skriv sedan:

```
scp (namnet på mappen med filerna i)/* (adressen till din server)
```

Det kan till exempel se ut så här:

```
scp flask_portfolio/* server1.google.se
```

Detta kommando kopierar allt i denna mapp till din server.

Logga nu in på din server med hjälp av ”ssh”-kommandot:

```
ssh (din serveradress)
```

När du väl är inne på servern, skriv:

```
tmux new -s (namn)
```

tmux är ett kommando som håller processer som du startar inom den igång även om du loggar ut från din server. Namnet du ska mata in är ett namn på processen som ”tmux” håller reda på. Det

namnet är använt när du vill stänga ner och starta upp din server, så sätt det som något enkelt att komma ihåg. Till exempel:

tmux new -s Flask_portfolio

Så när du har skapat din process i ”tmux”, skriv:

python flask_portfolio.py

Se till att du är i den mappen där du kopierade filerna innan. Du kan använda dig av kommandot ”ls” för att se alla mappar och filer i den mappen du är just nu. Se till att ”flask_portfolio.py” är med i listan som kommer upp om du skriver in ”ls”.

Avsluta ditt tmux-kommando genom att trycka på Ctrl + B och sedan D. Du kan nu logga ut från din server och ha din hemsida vara kvar uppe efter att du loggar ut.

Om det inte fungerar så kan du (i servern) skriva:

tmux ls

Det kommandot listar alla dina kommandon som körs av ”tmux” medans du är borta. Om du hittar ditt kommando i den listan så är hemsidan uppe och körs.

4. Lägg till eller ta bort projekt efter att du har startat din server

När du vill lägga till ett projekt efter du startat din server så uppdaterar du ”data.json”-filen som du har på din dator (se steg 2).

4.1. Testa att allt fungerar på din hemdator

Detta steg är helt frivilligt, men starkt rekommenderat. Detta steg kräver att du har installerat Flask på din egen dator (se steg 1.1 och 1.1.1).

Navigera till mappen där du har din ”flask_portfolio.py” via terminalen. Kör den sedan:

python flask_portfolio.py

En lokal server kommer då startas på adressen som kommer upp i terminalfönstret. Skriv in den adressen i din browsers URL-fält (t.ex. ”127.0.0.1:5000”). Där ser du då hur allt kommer se ut om du laddar upp din JSON-fil på din server. Om hemsidan säger att något är fel, så kolla att du har skrivit rätt i din JSON-fil (t.ex. kan du ha glömt ett kommatecken eller liknande).

När du känner dig nöjd, tryck Ctrl + C i ditt terminalfönster för att stänga ner den lokala servern.

4.2. Lägg upp din JSON-fil på din hemsida

När du känner att din JSON-fil är redo att laddas upp på din server, navigera till mappen där du har din JSON-fil via terminalen och skriv in:

scp data.json (din serveradress)

Till exempel:

scp data.json server1.google.se

Din hemsida kan eventuellt sluta fungera medan din JSON-fil blir uppladdad. Men när den väl är uppladdad ska hemsidan fungera som vanligt.

5. Stänga ner servern

Om du av någon anledning skulle vilja stänga av din server så loggar du först in på din server:

ssh (din serveradress)

Efter det så kan du lista dina processer som kommandot ”tmux” håller på att köra automatiskt på servern:

tmux ls

I den listan som kommer upp får du ett namn för din Flask-process. Använd det namnet när du skriver:

tmux kill-session (namn)

Till exempel:

tmux kill-session Flask_portfolio

Detta stänger då av din Flask-server. Efter det är gjort kan du bara logga ut med hjälp av ”logout”-kommandot.

6. Felhantering

Om error-sidan skulle, vid någon gång du använder systemet, komma upp, så kan du se var det gick fel i koden genom att kolla i loggen. Loggen ligger i samma mapp som alla andra filer i systemet, men den skapas inte för ens systemet gör sin första loggning Det kommer göras så fort en användare går in på någon sida på hemsidan när den väl har lagts upp (eller om du själv går in medans du testar).

Loggarna läggs upp på så sätt att tiden ner till sekunden skrivs upp, följt av i vilken funktion i koden det gick fel, och därefter ett meddelande om vad systemet tyckte gick fel. T.ex. kan det stå:

”2014-10-16 08:17:13 | LOAD: Trying to load datafile 'data.json'”

”2014-10-16 08:17:13 | LOAD: Loaded datafile 'data.json'”

Denna loggningssekvens visar då att systemet kunde läsa din databas (filen 'data.json', se ovan). Om det däremot skulle stå:

”2014-10-16 08:17:13 | LOAD: Failed to load datafile 'data.json'”

Så betyder det att programmet inte kunde läsa din databas. Då borde du t.ex. kontrollera att ”data.json”-filen ligger i rätt mapp (samma mapp som logg-filen), eller att du har formaterat allt rätt (se punkt 4).