Project: Mörtfors buss applikation av Felix Morau & Victor Persson

I grunden har vi har använt oss av programmeringsspråket [Python](https://www.python.org/) för att bygga vår bokningsapplikation. (*version: Python 3.6*). För att arbeta mot själva databasen för att hämta och lagra data i vår Python kod har vi tagit hjälp av biblioteket “[psycopg2](http://initd.org/psycopg/)”.

Utöver detta har vi arbetat med [PostgreSQL](https://www.postgresql.org/) för att hantera databasen på samma vis som vi lärt oss i kursen. Med hjälp av dessa programmeringsspråk, bibliotek och frågespråk har vi byggt en backend som via terminal/cmd låter en användare boka bussresor via ett fiktivt bussbolag.

Applikationen använder sig inte av något särskilt frontend utan körs direkt i terminalen/cmd, men funktionaliteten mellan databasen och programmet fungerar minst lika väl. Det hela består i huvudsak av tre filer. Ett SQL script som ansluter sig till Malmö universitets databas och sedan listar alla tables som kan ses nedan i relationsmodellen, tillsammans med några INSERT satser för att fylla databasen.

De två andra filerna utgör själva applikationen och består av ett interface för kunder som vill registrera, boka och visa sina möjliga resor. Den andra python filen är en admin\_panel endast avsedd för de anställda på bussbolaget, här kan man registrerar nya chaufförer, turer samt uppdatera/lägga till chaufförer i efterhand

Installations guide för python & psycopg2 finns på [GitHub](https://github.com/VictorPersson/db-project-mortfors)

Funktioner i bus.py

Först importerar vi biblioteken psycopg2 som nämns ovan, plus random som är ett python standardbibliotek för att bland annat slumpa fram slumpmässiga integers som vi använder till många primärnycklar. Vidare ansluter vi oss till Malmö universitets databas genom en ‘conn’ med inloggnings variabler till DB servern **pgserver.mah.se.** Denna del av funktionen förekommer i alla våra funktioner då vi inte kunde få den globala att fungera på ett önskvärt vis.

**def busbooking():** är vår main funktion som kör alla andra funktioner utifrån olika if statments som beror på vad användaren väljer som input i menyn.

**def welcome():** Printer ett välkomstmeddelande till användaren

**def menu():** Printar visuellt de alternativ som användaren kan välja mellan ifrån main funktionen busbokning()

**def register\_traveller():** Först ansluts man till databasen.Sedan registrerar man sig som ny kund genom att svara på flera input frågor som genom cursor.execute(INSERT) lagras i vår databas. Primärnyckeln personid skapas genom randint från det importerade random biblioteket och blir ett värde mellan 1-100000000. Här skulle man kunna bygga en funktion som gör att den loopas om i fall att den skapar och lägger in ett värde som redan finns i db, men det är ju i nuläget 5 på miljonen.

**book\_trip():** I denna funktion får användaren på samma vis som i register\_traveller svara på en mängd inputs om vem man är genom what\_traveller, som sedan genom en cursor.execute(SELECT) sats letar igenom alla rader i tabellen Resenär där det namn man anger stämmer med det man angivit. Efter detta hämtas detta namns PersonID ut till python funktionen genom cursor.fetchone(). Detta ihop men vad användaren anger får destination fungerar på samma vis där python hämtar ut ett resmål användaren anger om det matcher en existerade tur ifrån tur tabellen. Det sista funktionen gör att är kolla om det antal säten som kunden vill boka finns tillgängliga eller inte, detta genom att hämta ut de platser som kunden har angivit från resan de vill boka och sedan genom en sum() funktion summerar detta. Sedan hämtas det in till python med fetchone(), efter detta subtraherar vi antalet som man angivet med det totala antalet säten. Genom en if/else sats kollar vi om det ens finns lagom många säten till den nuvarande kunden, om det finns så läggs man in i boking med sin PersonID, vilken ReseID man ska åka samt hur många platser.

**search\_trips():** Här ställs man till två val genom en if sats. Antigen kan man med en SELECT \* sats se alla resmål som nuvarande finns inlagda. Eller så man kan trycka 2 i menyn för att söka efter ett specifikt resmål genom en cursor.execute(SELECT) med en LIKE sats där det ord man skriver in sedan fetchas och visas för användaren ( ‘% sökord %’ )

**def spacer():** Printar en horisontell linje för att göra det enklare att läsa i terminalen.

Funktioner i admin\_buss.py

**def welcome():** Printer ett välkomstmeddelande till den anställde / admin

**def menu():** Printar visuellt de alternativ som admin kan välja mellan ifrån main funktionen admin\_panel()

**def register\_driver():** Enkel funktion där man kan lägga till nya chaufförer till Chaufför tabellen genom en cursor.execute(INSERT) sats som lägger in det admin skriver som inputs.

**def add\_route():** Add route fungerar väldigt likt register\_traveller(). Här får admin skriva in all information som ska fyllas i tabellen Tur genom in inputs. ReseID får vi genom att slumpa en integer mellan 1-100000000. Detta som vanligt genom en cursor.execute(INSERT) till Tur tabellen.

**def add\_driver\_later():** En funktion som tillåter admin att lägga till exakt vilken chaufför som ska göra vilken rutt, efter att man lagt till en Tur. Detta genom att hämta alla Turer med en SELECT \* execute. Sedan Hämtar man in alla och listar den, admin får senda ange vilken Tur\_ID hen vill lägga in / ändra chaufför i genom att ange chaufförens C\_ID. Om man anger ett existerade C\_ID ifrån Chaufför tabellen uppdateras Tur tabellen via en cursor.execute(update) funktion. Om inte har man inte angivit en existerande chaufför eller Tur\_ID och man får börja om.

**def spacer():** Printar en horisontell linje för att göra det enklare att läsa i terminalen.

Användarinstruktioner till Mörtfors bus bokningssystem

Installations guide för python & psycopg2 finns på [GitHub](https://github.com/VictorPersson/db-project-mortfors)

1. Gå till denna [GitHub](https://github.com/VictorPersson/db-project-mortfors) länk och ladda ner/klona projekt filerna.
2. Om du sitter på Mac eller Linux, starta din terminal, för Windows starta cmd
3. Navigera dit du laddade ner filerna, öppna bus.py och admin\_bus.py med valfri textredigerare
4. Vid rad 5 respektive rad 7, ta bort och lägg till dina egna personliga inloggningsuppgifterna till Maus databas.
5. Tillbaka till terminalen/cmd. Navigera till den plats där du laddade ner och sparade filerna.
6. Starta applikationen genom att skriva för att starta applikationen för användare python bus.py
7. Efter detta fungerar applikationen genom inmatning av knappval. Tryck 1 för att registrera dig som kund. Tryck 2 för att boka en resa, tryck 3 för att lista alla tillgängliga resor eller 4 för att stänga applikationen.
8. Vill du sedan testa applikationen som anställd så stänger du processen med Ctrl + C eller genom att trycka 4 i menyn. Sedan skriver du python admin\_bus.py
9. Även admin panelen fungerar efter samma principer, genom knappval i en meny som förklarar vad du ska ange, till exempel “Enter name:” osv

Databas schema (relationsmodellen)

* **Resenärer**: Där man registrerar som som användare (Primary Key: PID)
* **Chaufför:** C\_ID = Chaufför ID, tabell där man registrerar nya chaufförer
* **Tur:** Tabell där man lägger till resor. Rese ID = Unikt ID för varje resa. (Primary Key: Rese ID)
* **Bokning:** Här lagras vem som bokat vilken resa och hur många platser. (Primary key & foreign key = ReseID)
* **Plats**: En tabell över vilka resmål som finns tillgängliga. (Primary key = hållplats )

**Resenär**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PersonID** | **Namn** | **Epost** | **Adesss** | **Telefon** |
| 001 | Victor | vp-96@hotmail.com | Lantmannagatan 3B | 0709240058 |
| 002 | Kall | karl.olsson@gmail.se | Segervångsgatan 2 | 0709232142 |

**Chaufför**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **C\_ID** | **Personnummer** | **Namn** | **Address** | **Hemtelefon** |
| 001 | 197604051663 | Sven Nilsson | Jonasvägen 3 | 042191019 |
| 002 | 198603156531 | Oscar Björk | Fågelgränd 1 | 070343232 |

**Tur**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Rese** ID | **C\_ID** | **Datum** | **Avång** | **Ankomst** | **Kostnad** | **Platser** | **Från** | **Till** |
| 164 | 002 | 091218 | 0800 | 0900 | 150 | 50 | Malmö | Berlin |
| 112 | 198603156531 | 020518 | 1500 | 2000 | 500 | 78 | Göteborg | Malmö |

**Bokning**

|  |  |  |
| --- | --- | --- |
| **ReseID** | **PID** | **Bokad.plats** |
| 001 | 164 | 6 |

**Plats**

|  |  |  |
| --- | --- | --- |
| **Land** | **Stad** | **Hållplats** |
| Sverige | Malmö | Konsum |
| Berlin | Tyskland | Das das |