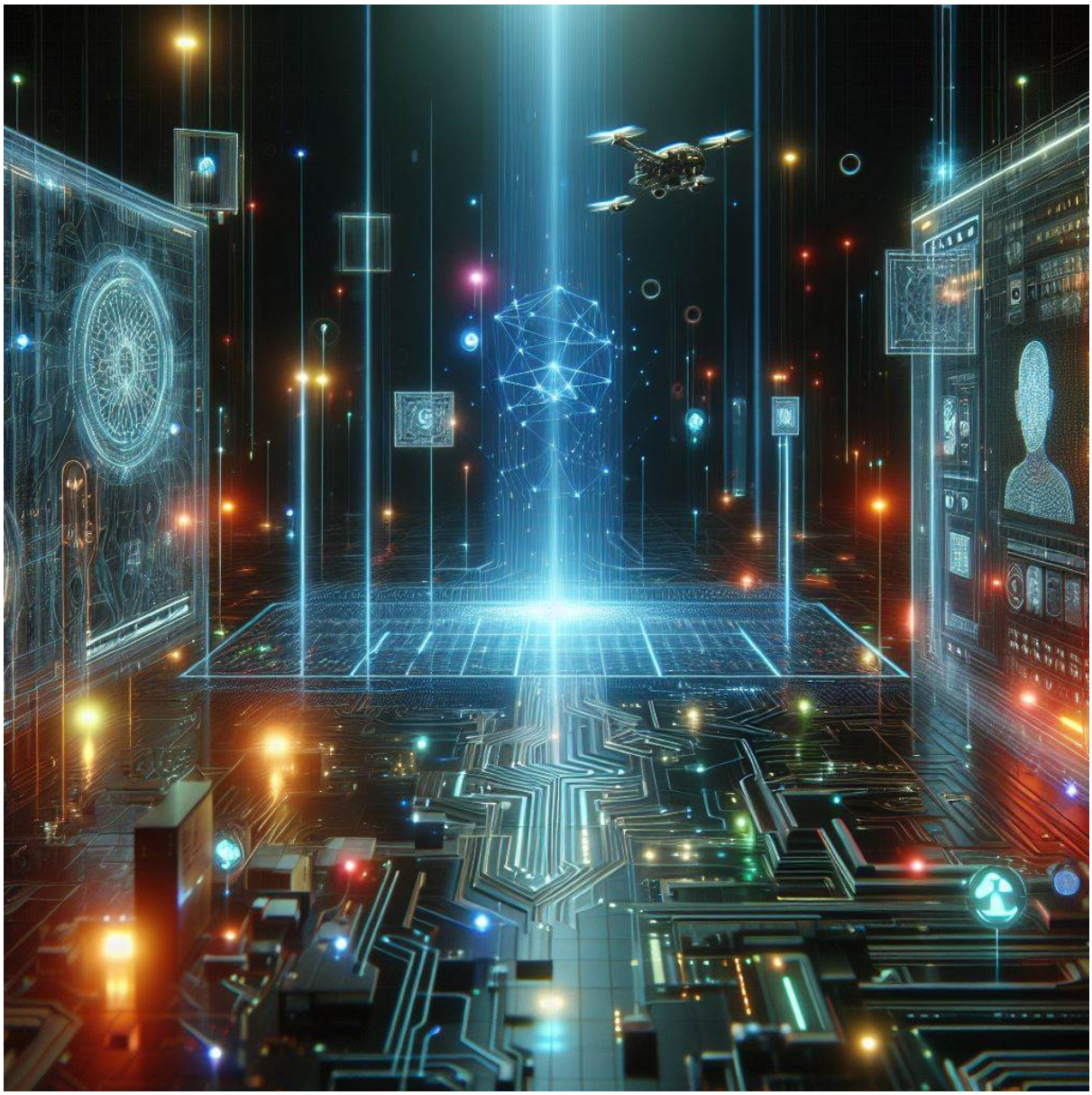

Figur 1: Cyberspace, bilden är genererad med hjälp av DALL·E 3.

Labb W1: Laboration i webbprogrammering

Denna uppgift testas inte på Kattis

1 Inledning

I denna laboration kommer du att utveckla en hjälpsam webbapplikation som interagerar med användaren via någon av webbläsarapplikationer Firefox eller/och Chrome för att användaren ska kunna skapa en inköpslista till sitt hushållet.

Observera att i denna labb är det du och en eventuell labbpartners uppgift att välja vilka datatyper som ska lagra datan, att namnge och utforma eventuella hjälpfunktioner, att komma på lösningsmetoder och till sist implementera dem. Det är ett brott mot hederskodexen att be personer utanför labblaget med hjälp om detta, eller att använda artificiella intelligenser. Listan med förbjudna artificiella intelligenser inkluderar men är inte begränsad till ChatGPT och GitHub Copilot.

2 Förberedelse och konfiguration.

I den här labben ska du använda **PHP** och databasen **SQLite** för lagring av data, men om du har erfarenhet av andra databaser så som PostgreSQL så är det ett bra alternativ. PHP-version 7.4.3 finns installerad på skolans datorer och det är färdigkonfigurerat med SQLite som databas. Om du inte vill använda skolans datorer så får du installera ovannämnda programvaror på den dator du kommer att använda för implementation men vi rekommenderar starkt att du använder skolans datorer för att minimera installationsproblem. För att du ska kunna använda skolans datorer behöver du göra följande:

1. Logga in på någon av ubuntu-datorer i labbsalarna och använd terminalen i resterande steg. (eller fjärrinlogga med ssh på servern `student-shell.sys.kth.se`)
2. Skapa en mapp i din hemkatalog med namnet `public_php`:

   ```
   mkdir ~/public_php
   ```

3. Sätt om afs-rättigheterna så att ingen annan afs-användare över hela världen kan komma åt dina filer, samt ser till att webbservern får läsrättigheter:
   `fs sa  /public_php system:anyuser none httpd-course read`
   Om du har en labbpartner behöver du sätta om rättigheterna så att hen kan läsa och modifiera filerna:

   ```
   fs sa ~/public_php [användarnamn_för_din_labbpartner] rlidw
   ```

4. Du behöver skapa en mapp för dina databasfiler låt oss anta att mappen heter `database_files`:
   `mkdir  /public_php/database_files`
5. Nu behöver du sätta om afs-rättigheter så att dina PHP-skript kan få tillgång till databasfilerna:

   ```
   fs sa ~/public_php/database_files httpd-course rliwd
   ```

Nu kan du lägga dina PHP-filer i mappen

```
~/public_php
```

och sedan via följande URL där du byter USERNAME mot ditt eget användarnamn kommer du kunna köra dina program:

`https://course-dd1366.eecs.kth.se/~USERNAME`

Databas-filen ska vara i mappen

`~/public_php/database_files`

Exempelvis filen welcome.php som finns i mappen

```
public_php
```

för användaren med användarnamnet vahid kommer att kunna köras genom följande länk:

`https://course-dd1366.eecs.kth.se/~vahid/welcome.php`

Hämta skripten `welcome.php`, `shopping_list.php` och placera i `public_php` samt hämta databasfilen `products.db` samt placera i mappen `~/public_php/database_files`
Kör skripten via url:ena:

`https://course-dd1366.eecs.kth.se/~[ditt_användarnamn]/welcome.php`

`https://course-dd1366.eecs.kth.se/~[ditt_användarnamn]/shopping_list.php`

3 Labbinnehåll och mål

Webbapplikationen ska kunna lista de varor som är slut eller snart kommer att vara slut i hushållet baserat på en analys av användarens konsumtionsvanor.

3.1 Generellt förfarande

1. En användare loggar in på inköpslistans webbapplikation.
2. Efter inloggningen dirigeras användaren till inköpslistan, där programmet automatiskt föreslår varor baserat på tidigare inköpsmönster.
3. Programmet analyserar historiska inköpsdata och genererar förslag baserat på genomsnittliga förbrukningsintervaller för varje produkt.
4. Användaren har möjlighet att manuellt lägga till varor i inköpslistan som inte automatiskt föreslagits av programmet.
5. Efter att ha granskat och modifierat inköpslistan klickar användaren på "Spara" för att generera en slutgiltig lista.
6. Den slutgiltiga listan lagras i databasen för att användaren ska kunna bekräfta sina inköp.
7. Användaren besöker bekräftelsesidan för att bekräfta alla inköp. Här går det även att göra ytterligare impulsköp.
8. Programmet uppdaterar databasen med nya inköpsdatum för de köpta varorna.
9. Användaren loggar sedan ut från applikationen.

3.2 Rekommenderade regler

Inköpsdatum är centralt för denna webbapplikation. De här reglerna kan kanske hjälpa att förstå logiken bättre:

1. En vara som finns i databasen men INTE har något inköpsdatum är en vara som aldrig köpts tidigare därför ska varan alltid listas med i inköpslistan.
2. När en vara köps så ska databasen uppdateras med den informationen, alltså ett nytt datum ska läggas till i databasen för just den köpta varan.
3. När en vara tas bort från databasen (via webbgränssnittet) då ska varan med all relaterade data tas bort från databasen.

4 Kravspecifikation

4.1 Krav på din webbläsare

Det är viktigt att din webbapplikation kan användas med Firefox och Chrome.

4.2 Krav på din databas

Databasen du använder för lagring av data måste vara en SQL-databas.

4.3 Nödvändiga sidor

Webbapplikationen behöver ha följande sidor:

1. Sida för att kunna skapa konto: Denna sida måste ha fält för användarnamn och lösenord.
2. Inloggningssida: För att användaren ska kunna logga in. Denna sida ska visas igen efter en misslyckad inloggning med information om att inloggningen blev misslyckad och användaren ska alltså försöka logga in igen på samma sida.
3. Sida för att kunna lägga till eller ta bort hushållsvaror i databasen.
4. Sida för att skapa/modifiera inköpslista.
5. Bekräftelsesida som visar själva inköpslistan som alltså används under den tid man är ute och handlar varor, varje vara ska sammanfogas med en knapp (t.ex checkbox) för att markera när man plockar in en vara i sin varukorg. Samt en knapp när man är klar med shoppandet.
6. Man ska kunna logga ut genom att klicka på en knapp som finns i alla sidor förutom inloggningssidan. Har man tryckt på utloggningsknappen då måste man logga in igen om man vill använda webbapplikationen.

4.4 Säkerhetskrav

När du utvecklar en webbapplikation, ska du ta hänsyn till säkerhetsaspekter för webben. Det är viktigt att inte lita på indata från webbklienten.

4.4.1 Databasintrång

En mycket allvarlig och känd säkerhetsrisk är SQL-injektion. Se
https://owasp.org/www-community/attacks/SQL_Injection
Ditt program ska alltid kontrollera att emottaget data från webbklienten inte skadar din webbapplikation och databasen.

4.4.2 Databaskonsistens och Integritet

För att säkerställa att databasen förblir i ett konsistent tillstånd även när flera användare interagerar samtidigt är det ytterst viktigt att använda transaktionshantering. Så kallade *race conditions* kan uppstå där vad som lagras i databasen beror på svårkontrollerade tidsberoende omständigheter. Det finns många sätt att undvika detta, t.ex. att inte tillåta två användare att vara inloggade samtidigt på ett och samma konto. En kan också tillåta att flera personer ska kunna logga in samtidigt och ta ut listor men ditt program ska ta hänsyn till att det kan vara flera personer som arbetar med varsin inköpslista. Du får välja vilket sätt du vill. Det första sättet är enklare dock har sina nackdelar. Kan du komma på vilka nackdelar vi tänker på?

4.4.3 Lösenordshantering

För att undvika att lösenorden läcker ut vid ett eventuellt intrång i servern får lösenorden inte lagras i klartext i databasen. Detta innebär att när ett konto skapas ska enbart ett hashvärde baserat på lösenordet sparas i databasen, och då en användare försöker logga in ska hashvärdet av det inmatade lösenordet jämföras med värdet i databasen. En lämplig hashfunktion är t.ex. SHA3-512.

4.4.4 Examineras inte i denna labb

Det finns även andra sårbarheter sådana som XSS, som inte är direkt farlig för webbapplikationen men utnyttjar webbapplikationens sårbarhet för att skada andra webbklienter som kommer i kontakt med webbapplikationen. Dessa lämnar vi till andra kurser som datasäkerhet.

5 Scenarier

5.1 Scenario 1

1. Användaren loggar in och efter inloggningen visas sidan med inköpslista.
2. Programmet föreslår de varor som efter analysen bör vara slut eller snart slut i hushållet. Programmets förslag är baserat på genomsnitt av intervallen av alla tidigare inköp av den varan samt när nästa inköp kommer att vara. T.ex från databasen kan man lista ut att det har gjorts inköp på mjölk följande datum: 2024-01-01, 2024-01-05, 2024-01-08, 2024-01-13
   Från datumen framgår att i genomsnitt så är intervallet mellan inköpen (4+3+5) = 4 dagar, vilket innebär om datumet då man genererar ny inköpslista som är den 17:e januari 2024 eller senare så ska mjölk finnas med som förslag i inköpslistan och annars inte.
   När det gäller andra varor sådana som lampa som har något längre förbrukningstid så kan detta intervallet vara fler månader, alltså programmet behöver beräkna genomsnitt intervall för varje vara som finns i databasen.
   Dessutom användargränssnittet ska ge möjligheten till användaren att välja varor som finns i databasen men inte finns i förslaget med inköpslistan . Så om man märker att mjölken är slut men programmet har det inte som förslag i inköpslistan så ska man kunna lägga till ”mjölk” i inköpslistan.
3. Användaren känner sig nöjd med listan och klickar på spara. Då genereras en slutlig lista. Listan ska lagras in databasen för att användaren ska bekräfta att alla varor var köpta. När man är ute och handlar så märker man att butiken har en ny vara som man inte har testat tidigare och skulle vilja testa varan. Då borde man kunna lägga till den nya varan i samband med bekräftelsesidan.
4. Användaren har handlat klart och bekräftelsesidan och bekräftar att alla varor är köpta. Då uppdaterar programmet databasen med nya inköpsdatum för alla köpta varor.
5. Användaren loggar ut.

5.2 Scenario 2

Detta scenario betonar användning av historisk dataanalys för att automatisera varuföreslagning och en smidig bekräftelseförfarande för att bekräfta och uppdatera inköp. Du kan anpassa och utöka scenariot enligt dina specifika laborationsmål och behov:

1. En användare loggar in på inköpslistans webbapplikation.
2. Efter inloggningen dirigeras användaren till inköpslistan, där programmet automatiskt föreslår varor baserat på tidigare inköpsmönster.
3. Programmet analyserar historiska inköpsdata och genererar förslag baserat på genomsnittliga förbrukningsintervaller för varje produkt.
4. Användaren har möjlighet att manuellt lägga till varor i inköpslistan som inte automatiskt föreslagits av programmet.
5. Efter att ha granskat och modifierat inköpslistan klickar användaren på “Spara” för att generera en slutgiltig lista.5.2.6. Den slutgiltiga listan lagras i databasen för att användaren ska kunna bekräfta sina inköp.

6. Användaren bekräftar alla inköp och kan även lägga till eventuell ytterligare varor som köptes spontant under shoppingrundan.

7. När användaren har bekräftat uppdaterar programmet databasen med nya inköpsdatum för de köpta varorna.

8. Användaren loggar sedan ut från applikationen.

5.3 Scenario 3

Detta scenario belyser processen för misslyckad inloggning vilket är en viktig funktion för användarhanteringssäkerhet och användarvänlighet. Du kan anpassa detaljerna efter din laborationskontext och önskemål.

1. En användare försöker logga in på inköpslistans webbapplikation.

2. Användaren anger felaktigt lösenord eller användarnamn och får ett meddelande om att inloggningen har misslyckats.

3. Användaren försöker igen, den här gången lyckas hen logga in.

4. Systemet visar användaren inköpslistan efter inloggningen.

5.4 Scenario 4

Detta scenario fokuserar på att stödja användaren för att kunna hantera situationer där en önskad produkt inte är tillgänglig längre (t.ex. leveransproblem), och istället för att helt ta bort den från inköpslistan. Då tillåts användaren att ersätta den med en liknande vara och behålla båda i databasen för framtida referens. Du kan anpassa detta scenario baserat på de specifika kraven och funktioner du vill betona i din laboration.

1. Användaren loggar in på webbapplikation för att uppdatera och genomföra inköp.

2. Användaren granskar inköpslistan och märker att en specifik vara, “Produkt A”, som normalt köps är inte tillgänglig i butiken.

3. I stället för att ta bort “Produkt A” från listan beslutar användaren att ersätta den med en liknande vara, “Produkt B”, som finns tillgänglig i butiken.

4. Användaren markerar “Produkt B” som en tillfällig ersättning för “Produkt A” direkt på inköpslistan.

5. Efter genomförda inköp och bekräftelse av listan läggs både “Produkt A” och “Produkt B” till i databasen.

6. Systemet registrerar den nya relationen och förstår att “Produkt B” temporärt ersätter “Produkt A” för detta inköp.

7. Vid framtida förslag kommer systemet att överväga både “Produkt A” och “Produkt B” som möjliga inköp baserat på användarens tidigare beteende.

8. Användaren loggar ut efter att ha genomfört sina inköp och uppdateringar.

5.5 Scenario 5

Detta scenario illustrerar hur användaren hanterar situationen där en produkt kommer att sluta erbjudas och hur systemet anpassar sig genom att utesluta den från framtida inköpsförslag. Du kan anpassa detaljerna och betoningen beroende på de specifika aspekter av systemet eller användarinteraktionen som du vill belysa i din laboration.

1. Användaren loggar in på inköpslistans webbapplikation för att hantera och uppdatera sin inköpslista.
2. Under granskning av inköpslistan upptäcker användaren att en specifik vara, “Produkt C”, inte längre kommer att vara tillgänglig i framtiden på grund av leverantörsproblem eller utgående produktion.
3. Användaren beslutar att ta bort “Produkt C” från inköpslistan och förhindra att den föreslås i framtida inköpsförslag.
4. Efter borttagningen visas en bekräftelsesida där användaren får information om att “Produkt C” inte längre kommer att ingå i framtida förslag.
5. Systemet uppdaterar databasen genom att ta bort “Produkt C”.
6. Vid nästa inköpsanalys inkluderar systemet inte längre “Produkt C” som ett förslag baserat på användarens konsumtionsvanor.
7. Användaren loggar ut efter att ha slutfört uppdateringen av inköpslistan.