



Datum: 2025-03-20

# Webbaserad Bibliotekshantering

Bashir - Mahmoud - Abdullahi Asad

---

<b>Datum:</b>	2024-01-18
<b>Författare:</b>	Basir - Mahmoud - Abdi
<b>Examinator:</b>	Darwin och Thomas
<b>Program:</b>	DSD400 (Databaser)
<b>Huvudområde:</b>	Datateknik
<b>Utbildningsnivå:</b>	Grundnivå
 <b>Utgivare:</b>	 Högskolan Väst, Institutionen för ingenjörsvetenskap 461 86 Trollhättan Tel: 0520-22 30 00 Fax: 0520-22 32 99, <a href="http://www.hv.se">www.hv.se</a>

---

## Inledning

Många biblioteker kämpar fortfarande med gamla systemet eller papperbaserade hantering av böcker. Det är ofta tidskrävande och inneaktivt. Genom att skapa ett modernt och digitalt bibliotekssystem blir det enklare att låna, reservera och hålla koll på böcker. Alltså det underlättar för besökare att följa sina låna.

## Bakgrund

Hantering av böcker på bibliotek är avgörande för att kunna hålla ordning på lån, reserveringar och böcker. Många bibliotek använder föråldrade system eller manuella metoder som kan vara ineffektiva och besvärliga. Tack vare teknologins framsteg finns nu möjligheten att skapa smidigare och mer användarvänliga lösningar som både bibliotekspersonal och besökare enkelt kan använda online.

Detta projekt fokuserar på att utveckla ett webbaserat biblioteks hanteringssystem som tillåter användare att boka och låna böcker direkt via en webbplattform. Systemet underlättar för besökare att följa sina lån och för bibliotekspersonal att administrera böcker samt användarinformation.

## Syfte

Målet med projektet var att bygga ett system som gör det enkelt att:

- Låna och reservera böcker online
- Se vilka böcker som finns tillgängliga
- Administrera bokinformation, lån och bokningar.

---

## Metod

Teknologi Val:

Frontend: Next Js(React) med Typescript

- Användningsbar för snabb laddning via server side och enkel att ansluta med superbase, alltså god utvecklar upplevelser.

Backend: Supabase

- Supabase är ett open-source alternativ till Firebase. Fördelen med det är att du har mer kontroll över din data och möjlighet till egen hostning.

Databas: PostgreSQL ( via Supabase)

- Bygga strukturerade tabeller för böcker, användare lån och reservationer. Det vill säga stabil och relationsbaserad databas som passar perfekt för bibliotekssystem.

## 2. Systemdesign

Användaren interagerar med Frontend : (NextJs) sedan API anropas Superbase, därefter lagrar data i PostgreSQL.

Databastabeller:

- Users: Spara information om användare, till exempel e-postadress och lösenord.
- Books: Innehåller bokinformation som titel på boken, bild och tillgänglighet.
- Borrow: Loggar vilka böcker som lånats av vem och när.
- Reservations: Sparar bokningar av böcker som är upptagna.

**Funktion i systemet:**

**För användare:**

- Skapa konto och logga in
- Låna böcker som är tillgängliga
- Se vilka böcker som du har lånat
- Returnera böcker
- Reservera böcker som redan är utlånade

Frontend byggdes med HTML, CSS och JavaScript för att skapa själva webbsidan. HTML ger strukturen, CSS styr hur sidan ser ut, och JavaScript gör sidan mer interaktiv.

**Backend (Server):**

---

Backend skapades med Flask. Flask hjälper till att skapa API:er som skickar och tar emot data mellan användarens webbläsare och databasen. Flask hanterar också användarinloggning och annan logik.

**Databas:**

Databasen lagrar all data om användare, böcker, lån och reserveringar. Jag valde MySQL för att det är en pålitlig och snabb databas som passar bra för den här typen av system.

**Databasstruktur:**

Databasen består av följande tabeller:

**Users:** Här lagras information om användare, till exempel deras e-postadress och lösenord.

**Books:** Här lagras all information om böcker, som titel, tillgänglighet och bild.

**Borrow:** Här registreras alla lån, det vill säga vilken bok som lånades av vilken användare och när boken ska lämnas tillbaka.

**Reservations:** Här lagras alla bokreserveringar, alltså böcker som användare har reserverat för framtida lån.

## 3. Beskrivning av Systemet

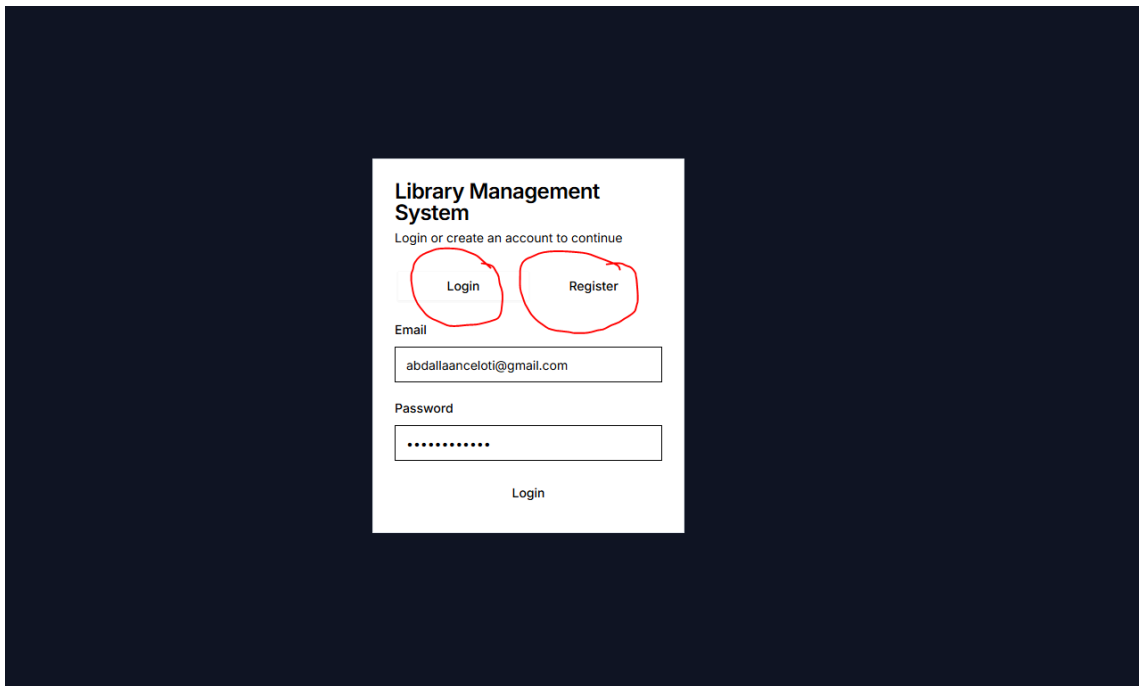
**Hur systemet fungerar:**

**För användare:**

Som användare ska du skapa ett konto och logga in för att komma åt bibliotekets dashboard där du har lista av alla böcker och andra funktionaliteter. När du är inloggad kan du:

- Låna böcker som är tillgängliga
- Se vilka böcker som du har lånat
- Returnera böcker
- Reservera böcker som redan är utlånade

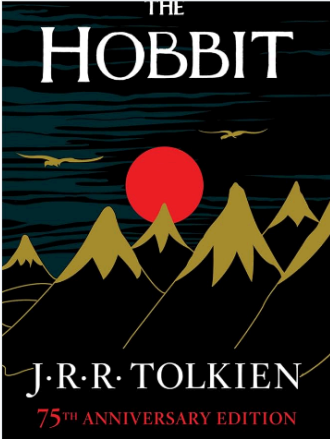
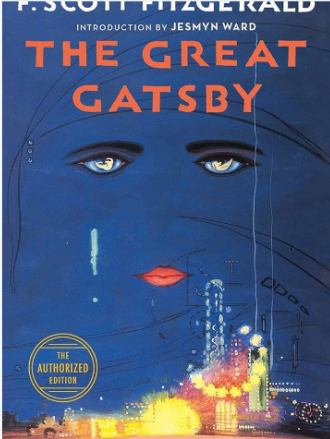
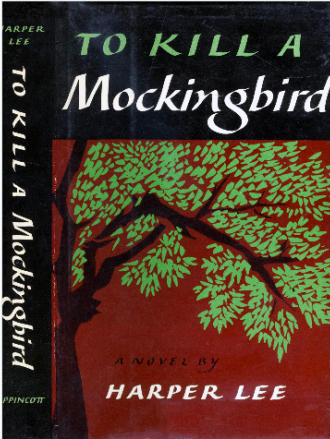
**Skärmdumpar:**



- Inloggningssida: Användare loggar in med e-post och lösenord eller registrerar ett konto sedan loggar in.

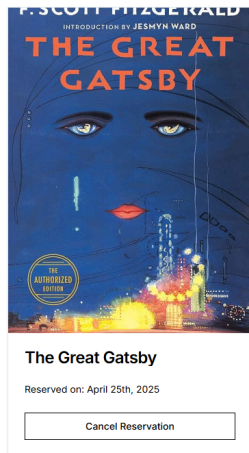
**Library System** [Books](#) [My Borrowed Books](#) [My Reservations](#)

### Available Books

		
<b>The Hobbit</b>	<b>The Great Gatsby</b>	<b>To Kill a Mockingbird</b>
Not Available	Not Available	Not Available
<input type="button" value="Reserve"/>	<input type="button" value="Reserve"/>	<input type="button" value="Reserve"/>

- Boklista: Visar alla böcker med bild och tillgänglighet

### My Reserved Books



- Reservation: Visar vilka böcker som du har reserverat.

## 4. Analys av Implementeringen

### Använda Teknologier:

**NextJS & Typescript:** Modern frontend med bra prestanda och tydlig kodstruktur.

Flask används för att bygga backend-API:erna som kommunicerar med frontend och databasen. Flask är lätt att använda och perfekt för mindre system som det här.

**Supabase & PostgreSQL:** Lätt att använda backend med realtids funktioner, hantera data och hosta.

**Utmaningar:** Samtidiga lån, alltså löses genom databastraktioner för att undvika konflikter. Och säkerhet, det vill säga lösenord lagras säkert, sessionhantering håller användaren inloggad.

### Utmaningar:

**Samtliga lån:** För att hantera flera användare som försöker låna samma bok samtidigt, alltså löses genom databas transaktioner för att undvika konflikter.

**Säkerhet:** Användarnas lösenord lagras aldrig i klartext- istället används kryptering för att skydda dem, det vill säga lösenord lagras säkert. Dessutom ser vi till att

---

sessionshantering fungerar smidigt så att användaren kan stanna inloggad utan att behöva upprepa inloggningen alltid. Håller användaren inloggad.

**Prestanda:**

**Effektivitet:** Systemet är optimerat för att fungera snabbt och stabilt, även när många användare är aktiva samtidigt. Genom effektiva databasanrop ger snabb respons och minskar laddningstiderna och säkerställer en bra användarupplevelse.

**Säkerhet:** Lösenord sparas aldrig i klartext, och sessionshantering gör att användare inte behöver logga in på nytt hela tiden.

## 5. Slutsatser

**Resultat:**

Projektet har resulterat i ett fungerande bok hanteringssystem. Användare kan enkelt låna och reservera böcker. Användaren kan också se vilka böcker som är tillgängliga, vilka som inte är tillgängliga.

## Analys

Systemet visade sig fungera stabilt och användarvänligt, vilket bekräftar att valet av moderna tekniker som superbase och NextJS passar utmärkt för den här typen av applikationer.

Analysera resultaten här:

- En bra modern och enkelt användningsbar design
- Systemet erbjuder enkla funktioner (låna, reservera och bokning)
- 

## 1 Slutsats

- Enkel och modern design: Gränssnittet är rent och intuitivt, vilket gör det lätt för användare att navigera och utföra uppgifter utan onödiga steg.
- Fokuserar funktionalitet: Systemet erbjuder de viktigaste funktioner som bokning och lån hantering på smidigt sätt.



- 
- Bra prestanda och tillförlitlighet: Tack vare en välstrukturerad backend och optimerade databasanrop svara systemet snabbt även vid ökad belastning.

Sammanfattningsvis är detta en effektiv och välfungerande lösning som uppfyller användarnas behov samtidigt som den bygger på modern och bra tekniker.

## Referenser

1. Supabase. (2023). Supabase Documentation: Building a fullstack app. Retrieved from <https://supabase.com/docs>
2. W3Schools. (2023). React Hooks Tutorial. Retrieved from [https://www.w3schools.com/react/react\\_hooks.asp](https://www.w3schools.com/react/react_hooks.asp)
3. Supabase crash Course: <https://www.youtube.com/watch?v=ydz7Dj5QHKY&list=PL4cUxeGkcC9hUb6sHthUEwG7r9VDPBMKO>
4. Databases Faster With Supabase: <https://www.youtube.com/watch?v=ueCECQ24STI>
5. Real-World open sources projects: <https://vercel.com/templates/next.js> and <https://github.com/vercel/next-learn>
- 6.