**Project 2 Planning**

**Library Management System**

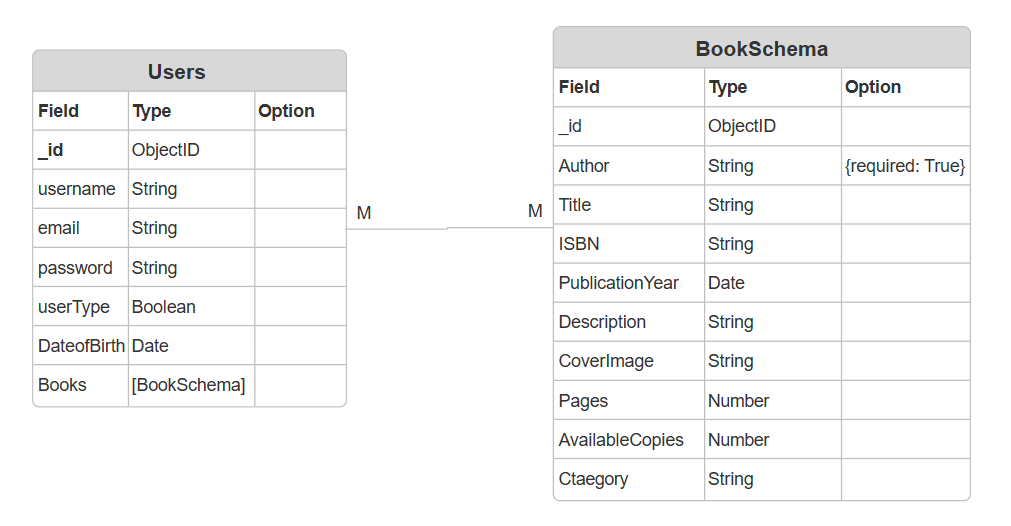
**Layla Jaffer Bader**

Library Management System

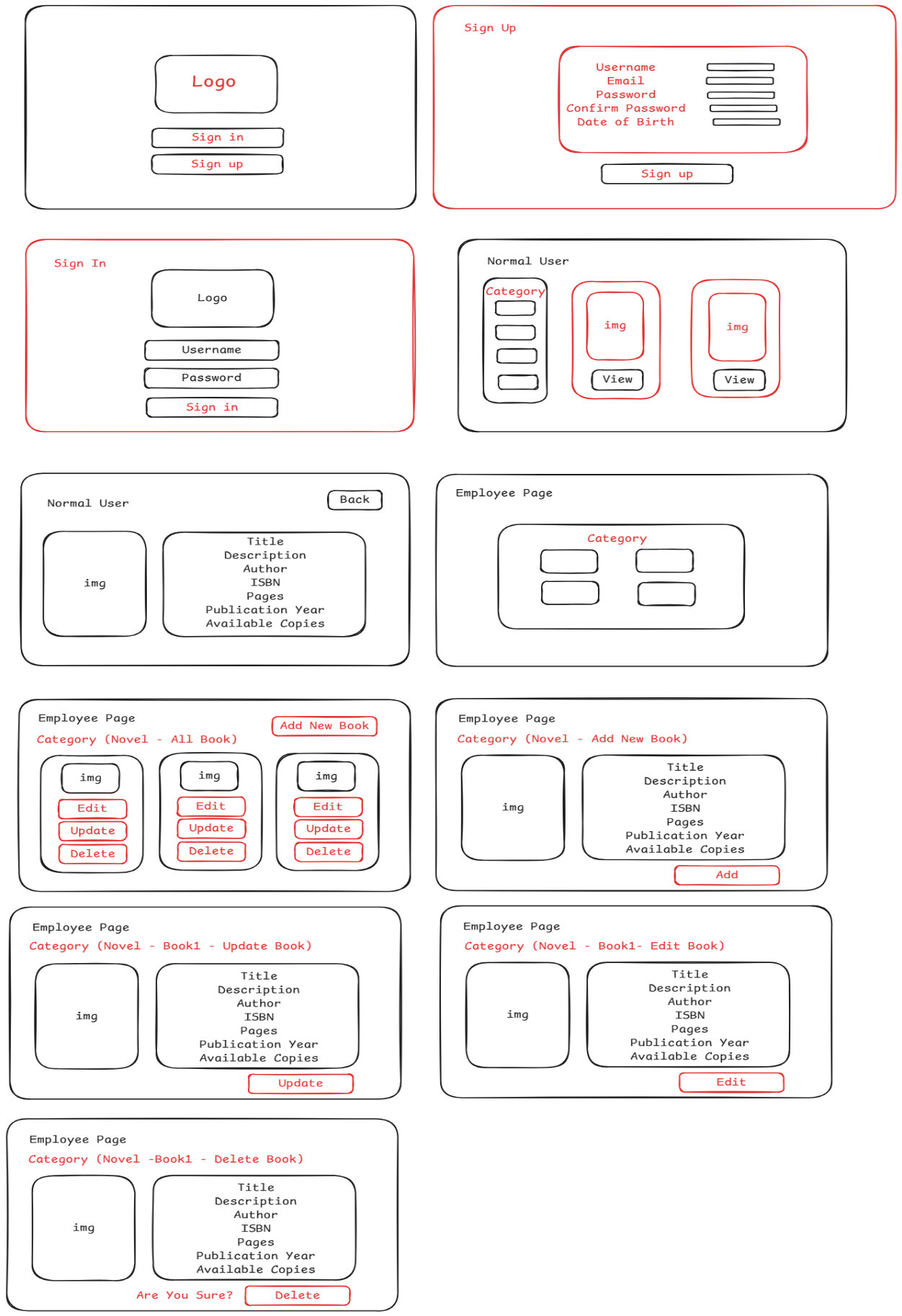
**Description**

This library management system requires both employees and normal users to register and log in. Employees can manage the library catalog, while normal users can only view the catalog without searching or filtering options.

**Data Models**



**Mock-UPS**



**User Story**

**Employee User Stories:**

1. As an employee, I want to log in to access catalog management features.
2. As an employee, I want to add new books to the library catalog with details such as title, author, ISBN, and available copies.
3. As an employee, I want to edit existing book information to keep the catalog updated.
4. As an employee, I want to delete books from the catalog when they are no longer available.
5. As an employee, I want to view a list of all books to manage and track their availability.

**Normal User Stories:**

1. As a normal user, I want to sign up and log in to access the library catalog.
2. As a normal user, I want to view a simple list of available books without a search or filtering option.
3. As a normal user, I want to click on a book to see its details (title, author, description, etc.).

**Pseudo Code for User Registration and Login**

**Authentication and Authorization**

// Check if user is authenticated

function authenticateUser(req, res, next) {

if (req.isAuthenticated()) {

return next();

}

res.redirect('/login');

}

// Check if user is an employee

function authorizeEmployee(req, res, next) {

if (req.user && req.user.userType === 'employee') {

return next();

}

res.status(403).send('Access Denied');

}

**Normal User Book Access (No Search)**

javascript

Copy code

// Normal user view (basic list of books)

app.get('/catalog', authenticateUser, async (req, res) => {

if (req.user.userType === 'normal') {

const books = await BookSchema.find({});

res.render('catalog', { books }); // Render basic list without search

} else {

res.status(403).send('Access Denied');

}

});

**Sign-Up, Login, and Dashboard Access**

// Register user

app.post('/register', async (req, res) => {

const { username, email, password, DateofBirth } = req.body;

const newUser = new User({ username, email, password, DateofBirth});

await newUser.save();

res.status(201).send('User registered successfully');

});

// Login user

app.post('/login', passport.authenticate('local', {

successRedirect: '/dashboard',

failureRedirect: '/login',

failureFlash: true

}));

// Dashboard access (only for authenticated users)

app.get('/dashboard', authenticateUser, async (req, res) => {

// Fetch and display book catalog for all users

const books = await BookSchema.find({});

res.render('dashboard', { books });

});