Programmer Documentation: Book Nook Web Application

**Introduction**

The **BookNook** is a simple dynamic web application developed as a term project for the Web Application Fundamentals (ZWA) course during the 2024/2025 academic year. The application is designed to manage a collection of books, allowing users to add, edit, view, and review books.

**Website Functionality:**

* **User Authentication:** Users can register, log in, and manage their profiles.
* **Book Management:**
  + **Add Books:** Authenticated users can add new books to the collection.
  + **Edit Books:** Users can edit details of existing books.
  + **View Books:** All users can browse and view details of books.
* **Reviews:**
  + **Add Reviews:** Authenticated users can add reviews to books.
  + **Edit Reviews:** Users can edit their existing reviews.
  + **View Reviews:** All users can read reviews associated with each book.
* **Administrative Functions:** Admin users have access to additional functionalities such as managing user profiles and viewing analytical charts.

Obsah obrázku text, snímek obrazovky, menu

Popis byl vytvořen automaticky**Project Structure and File Descriptions:**

The project comprises several directories and files, each serving a specific purpose:

* **Root Directory Files:**
  + index.php: The homepage of the application.
  + login.php: Handles user login functionality.
  + register.php: Manages user registration.
  + books.php: Displays the list of books.
  + book\_add.php: Form for adding new books.
  + book\_edit.php: Form for editing existing books.
  + book\_detail.php: Displays detailed information about a specific book.
  + reviews.php: Lists reviews for a specific book.
  + review\_edit.php: Form for editing a user's review.
  + profile.php: Displays user profile information.
  + profile-edit.php: Form for editing user profile.
  + admin.php: Admin dashboard for managing the application.
  + charts.php: Displays analytical charts (e.g., number of books, reviews).
* **Directories:**
  + css/: Contains stylesheets for the application's appearance.
  + js/: Houses JavaScript files for client-side interactivity.
  + images/: Stores image assets used in the application.
  + uploads/: Directory for user-uploaded files, such as book cover images.
  + src/: Contains PHP source files for various functionalities.
  + api/: Includes API endpoints for AJAX requests.
  + docs/: Documentation related to the project.

**Architecture:**

The application follows a simple PHP-based architecture without the use of external frameworks. It employs a modular structure, separating concerns across different files and directories.

* **Front-End:**
  + **HTML/CSS:** Defines the structure and styling of the web pages.
  + **JavaScript:** Adds interactivity and handles client-side form validation.
* **Back-End:**
  + **PHP:** Handles server-side logic, including form processing, database interactions, and session management.
  + **MySQL:** Used for data storage, managing information about users, books, and reviews.

**Form Handling:**

Forms are used throughout the application for user input, such as registration, login, adding books, and submitting reviews. Each form includes client-side validation using JavaScript and server-side validation in PHP to ensure data integrity. Upon submission, form data is sanitized and validated before being processed or stored in the database.

**Security Considerations:**

* **Input Validation and Sanitization:** All user inputs are validated and sanitized to prevent SQL injection and cross-site scripting (XSS) attacks.
* **Password Hashing:** User passwords are hashed using secure algorithms before storage to protect user credentials.
* **Session Management:** Sessions are used to manage user authentication states, with measures in place to prevent session hijacking.
* **Access Control:** Certain pages and functionalities are restricted based on user roles (e.g., admin vs. regular user) to ensure proper authorization.

**Database Schema:**

The database consists of tables corresponding to users, books, reviews, and other relevant entities. The database.sql file in the project contains the SQL statements necessary to create the required tables and relationships.

**Deployment:**

The application can be deployed using Docker, as indicated by the presence of Dockerfile and docker-compose.yml files. These files define the necessary environment configurations for running the application in a containerized setup.

**Detailed programmer-oriented explanation** of the files and their roles within the project structure:

**Root Directory Files**

1. **index.php**
   * **Purpose:** The homepage of the application. It acts as the entry point, displaying a summary of available features or a welcome message.
   * **Details:**
     + Displays a list of recent books or featured content.
     + Provides navigation links for authentication, book management, and user account management.
2. **login.php**
   * **Purpose:** Handles user authentication.
   * **Details:**
     + Presents a login form for users to input their credentials (email/username and password).
     + Implements server-side validation and redirects authenticated users to the dashboard or homepage.
     + Uses PHP sessions to manage user authentication states.
3. **register.php**
   * **Purpose:** Allows users to create an account.
   * **Details:**
     + Provides a registration form with fields such as username, email, and password.
     + Validates input data (e.g., checking for duplicate usernames or valid email formats).
     + Hashes passwords using a secure hashing algorithm before storing them in the database.
4. **books.php**
   * **Purpose:** Lists all books in the database.
   * **Details:**
     + Fetches book data from the database.
     + Displays book titles, authors, and optionally covers.
     + Includes search and filtering options for better navigation.
5. **book\_add.php**
   * **Purpose:** Provides a form for adding new books.
   * **Details:**
     + Includes fields for book title, author, genre, description, and cover image upload.
     + Handles file uploads securely, saving the image in the uploads/ directory.
     + Validates and sanitizes input before inserting data into the database.
6. **book\_edit.php**
   * **Purpose:** Allows users to modify book details.
   * **Details:**
     + Retrieves current book details from the database.
     + Pre-fills a form with existing data for user modification.
     + Updates the database with new values after validation.
7. **book\_detail.php**
   * **Purpose:** Displays detailed information about a single book.
   * **Details:**
     + Shows book title, author, description, genre, and reviews.
     + Includes links or buttons for adding or editing reviews.
8. **reviews.php**
   * **Purpose:** Lists reviews for a specific book.
   * **Details:**
     + Fetches reviews from the database related to a given book ID.
     + Displays review text, rating, and the reviewer's username.
     + Allows navigation to add or edit reviews (if the user is authenticated).
9. **review\_edit.php**
   * **Purpose:** Allows users to modify their own reviews.
   * **Details:**
     + Retrieves the user's existing review for a specific book.
     + Provides a form for editing review content and ratings.
     + Updates the database after validation.
10. **profile.php**
    * **Purpose:** Displays user profile information.
    * **Details:**
      + Fetches and shows user-specific data like username, email, and recent activity.
      + Offers a link to edit the profile.
11. **profile-edit.php**
    * **Purpose:** Allows users to update their profile.
    * **Details:**
      + Provides a form to change profile information such as username and email.
      + Validates input and updates the database accordingly.
12. **admin.php**
    * **Purpose:** Serves as an admin dashboard.
    * **Details:**
      + Provides tools for managing users and books.
      + Displays statistics like users, and reviews.
      + Includes options for deleting or modifying user accounts and reviews.
13. **charts.php**
    * **Purpose:** Displays analytical data visually.
    * **Details:**
      + Uses data from the database to create charts (e.g., number of books by genre, reviews per book).

**Directories**

1. **css/**
   * Contains all stylesheets.
   * Files define the visual appearance and layout of the website, ensuring consistent design.
2. **js/**
   * Contains JavaScript files that handle client-side interactivity.
   * Includes functions for form validation, AJAX requests, and interactive elements like modals.
3. **images/**
   * Stores static image assets, such as icons, logos, or background images.
4. **uploads/**
   * Serves as a storage directory for user-uploaded files, like book cover images.
   * Security measures include validation of file types and naming conventions to prevent overwrites or malicious uploads.
5. **src/**
   * Contains PHP scripts that define core functionality.
   * Includes reusable modules such as database connection files, utility functions, and middleware for session handling.
6. **api/**
   * Contains PHP files for handling AJAX requests.
   * Facilitates asynchronous operations like dynamic data loading or real-time validation.
7. **docs/**
   * Houses project-related documentation.

**Additional Notes**

1. **Database Interaction**
   * Each file dealing with dynamic content interacts with the database using SQL queries.
   * Prepared statements are used to prevent SQL injection.
2. **Security**
   * Files managing sensitive operations (e.g., login.php, register.php) enforce strict input validation and hashing algorithms for password management.
   * Access to admin-specific files like admin.php is restricted using role-based checks.
3. **Error Handling**
   * Each PHP file includes error-handling mechanisms to manage issues such as failed database connections or invalid form submissions.