אתר onlinebookstore

מסמך תכנון

מגישים:

ניר טוביה

דוד מיאסקובסקי

אילון ריינשטין

תיאור מסמך:

תאור כללי:

* מינוחים בשימוש מסמך
* טכנולוגיות בשימוש המערכת
* העלאת מערכת
* נקודות כניסה למערכת
* בסיס הנתונים
* גישה לבסיס הנתונים
* הטבלאות בבסיס הנתונים
* ERD
* תרשים ככלי של המערכת
* ספריות JAVA עיקריות בשימוש

דפי הhtml.

חבילת controller:

* AdminController
* AuthenticationController
* BookController
* CartController
* HomeController
* OrderController
* UserController

חבילת model.

חבילת repository

חבילת service:

* BookService
* CartService
* OrderService
* UserService

קבצים חשובים נוספים.

תיאור כללי:

מסמך זה בא לתאר את מבנה המערכת שמרכיבה את אתר "online book store" ואת הטכנולוגיות שבשימוש.

מינוחים בשימוש במסמך:

admin – משתמש באתר מסוג מנהל אשר מוסיף ומעדכן ספרים ומלאי.

Member – משתמש במערכת שהוא קהל היעד של המערכת אשר מתחבר כדי לקנות ספרים

Guest – משתמש זר שיכול ליצור משתמש ולהתחבר אליו, או להתחבר למשתמש קיים

טכנולוגיות בשימוש המערכת:

אתר האינטרנט online book store משלב בתוכו מספר טכנולוגיות:

* H2 Database – מכיל את טבלאות הנתונים של המערכת
* JDBC – מאפשר גישה לבסיס הנתונים
* Java classes, servlets – קוד Java שרץ בצד השרת
* JSP – עמודי HTML משולבים בקוד JAVA ומחוללים עמודי HTML לצד לקוח.
* Spring Boot – כלי שעוזר בבניית web application ושירותים עם Spring Framework
* Maven+JRE – שרת web אשר מריץ את ה-servlets ועובד על Tomcat.
* Thymeleaf – מנוע צד שרת של Java שיכול לעבד templates HTML.

העלאת מערכת:

כדי לעבוד עם המערכת, צריך להתקין JDK 17 או 19. אחרי זה, ניתן להריץ את התוכנה עם הקובץ JAR המצורף דרך cmd.

נקודות כניסה למערכת:

הגישה למערכת מתבצעת ב HTTP מול שרת הmaven אשר מאזין לפורט 8080 במחשב מקומי.  
הגישה בעזרת הלינק הבא: <http://localhost:8080/>  
חיבור זה היינו חיבור כללי לכל המשתמשים, כי המערכת דורשת התחברות.  
ניסיון בלהתחבר לכל לינק אחר של האתר ללא התחברות תעביר אותנו לעמוד ה- login.

בסיס הנתונים:

במערכת זו ישנו בסיס נתונים השומר בתוכו הן נתונים סטטיים המוכנים מראש והן נתונים דינאמיים אשר נקלטים מהמשתמשים. בסיס הנתונים ממומש בעזרת H2 console server, המאזין על IP/TCP פורט ברירת המחדל 3306, והגישה אליו נעשית בעזרת התקן של H2 JDBC.

גישה לבסיס הנתונים:

התוך קובץ application properties אנחנו יוצרים מאגר נתונים הנקרא OnlineGrocerySystemDB ומתחברים אליו בעזרת התחברות אוטומטית של spring boot. ב- bookorderingsystem אנחנו מאתחלים את מאגר הנתונים ומוסיפים פריטים חסרים.

הטבלאות בבסיס הנתונים:

|  |  |
| --- | --- |
| שם טבלה | הגדרת טבלה |
| Book | מכיל את כל הספרים והפרטים שלהם. ה-admin מוסיף ספרים, עורך ספרים קיימים, ומוחק אותם. Users יכולים לעיין ו- members יכולים ועשות הזמנות מפתח: id |
| Order | מכיל את כל ההזמנות של ה-members. members יכולים לעשות הזמנה ולעיין בהזמנות שלהם. ה-admin יכול לעיין בהזמנות של כולם  מפתח: id, מחובר ל-Book עם ה-ID של Book |
| OrderItem | טבלת עזר שעוזרת לשמור את מספר הספרים שהוזמנו בכל הזמנה מכל סוג.  מפתח: id |
| Role | מכיל את ה- Role של משתמש. האם הוא member או admin. מפתח: id |
| User | מכיל את רשימת המשתמשים, כולל ה-admin וה-Members. מכיל פרטי משתמש, כולל שם משתמש וסיסמא. המשתמשים מחוברים ל-Role דרך מפתח. מפתח: id. מחוברים ל-Order לפי id. מחוברים ל-Role לפי id. |

|  |  |  |
| --- | --- | --- |
| Role | | |
| ID | PK |
| Name |  |

|  |  |
| --- | --- |
| ORDERS | |
| ID | PK |
| orderDateTime  books  totalPrice |  |

|  |  |
| --- | --- |
| BOOK | |
| ID | PK |
| name  description  price  author  stock  quantityToBuy  deleteStatus |  |

ERD:

|  |  |
| --- | --- |
| User | |
| ID | PK |
| firstname  lastname  username(Unique)  email(Unique)  password  address  Role  orders |  |

|  |  |
| --- | --- |
| ORDER\_ITEMS | |
| ID | PK |
| Order  Book  Quantity |  |

תרשים כללי של המערכת:

Repository Class Extending CRUD Operations

Maven Web Server

Model

Service Layer

Controller

IE Client

H2 Database

תמונה שמכילה טקסט, תרשים, תוכנית, שרטוט טכני

התיאור נוצר באופן אוטומטיUML:

ספריות JAVA עיקריות שבשימוש:

|  |  |  |
| --- | --- | --- |
| שם ספרייה | תאור |  |
| SpringFramework | ל-SpringFramework יש dependencies המספקים לנו ספריות עזר שעוזרות לנו במספר תחומים, כולל התחלת הפרויקט, יצירת קשר עם מסד נתונים, אבטחת מידע, אימות משתמש, ואינטרקציה עם ה-front. | spring-boot-autoconfigure:2.5.12  spring-boot-starter-aop:2.5.12  spring-boot-starter-data-jpa:2.5.12  spring-boot-starter-jdbc:2.5.12  spring-boot-starter-json:2.5.12  spring-boot-starter-security:2.5.12  spring-boot-starter-test:2.5.12  spring-boot-starter-thymeleaf:2.5.12  spring-boot-starter-tomcat:2.5.12  spring-boot-starter-validation:2.7.2  spring-boot-starter-web:2.5.12  spring-boot-starter:2.5.12  spring-boot-test-autoconfigure:2.5.12  spring-boot-test:2.5.12  spring-boot:2.5.12  spring-data-commons:2.5.10  spring-data-jpa:2.5.10  spring-security-config:5.5.5  spring-security-core:5.5.5  spring-security-crypto:5.5.5  spring-security-test:5.5.5  spring-security-web:5.5.5  spring-aop:5.3.18  spring-aspects:5.3.18  spring-beans:5.3.18  spring-context:5.3.18  spring-core:5.3.18  spring-expression:5.3.18  spring-jcl:5.3.18  spring-jdbc:5.3.18  spring-orm:5.3.18  spring-test:5.3.18  spring-tx:5.3.18  spring-web:5.3.18  spring-webmvc:5.3.18 |
| Com.H2database | ספרייה שמאפשרת שימוש בשרת H2 | h2:1.4.2002 |

דפי ה- HTML:

האתר שלנו מוצג בעזרת דפי HTML הבאים:

|  |  |
| --- | --- |
| שם קובץ HTML | תיאור קובץ |
| admin\_dashboard | זהו עמוד ראשי עבור משתמש מסוג admin שנותן אפשרות ל- admin לערוך, למחוק, ולחפש ספרים. יש ממנו גם גישה לחלקים אחרים באתר, והוא מוגן ככה שמשתמש ללא הרשאה לא יכול להיכנס לעמוד זה.   * כאשר עושים חיפוש, אנחנו משתמשים בקוד javascript שמובנה בתוך העמוד, והוא מסננן לנו את התוצאות לפי מה שהוזן. * בלחיצה על כפתור delete מופעל deleteProductById ב- bookcontroller. זה מסמן את הספר בתור מחוק, אבל משאיר אותו במסד הנתונים. * בלחיצה על כפתור update, אנחנו מועברים לעמוד update\_product עם פרטי הספר. |
| checkout | העמוד שמראה שההזמנה בוצע בהצלחה ומראה את פרטי ההזמנה.   * פרטי ההזמנה מוזנים בשכבת ה-controller. |
| customer\_dashboard | העמוד הראשי של member, שבוא ניתן לצפות בספרים ולהוסיף ספרים לרשימת קניות.   * כאשר עושים חיפוש, אנחנו משתמשים בקוד javascript שמובנה בתוך העמוד, והוא מסננן לנו את התוצאות לפי מה שהוזן. * בלחיצה על כפתור add to cart, יופעל addToCart ב- cartController שמוסיף כמות ספרים מסוימת מעל 0 ולא יותר מכלל הספרים במאגר הנתונים. * קודם יופעל bookRepository.findById ששולף את הספר לפי ה-id, ואז בודק האם הספר קיים וגם שהכמות במלאי לא קטנה מדי. אם לא קיים או יש מעט מדי ספרים, מחזיר out of stock. * אם הכמות מתאימה, אז יופעל cartService.addToCart, שמוסיף את הספר והכמות שלו לסל הקניות. * לאחר מכן, שולח אותנו לעמוד shopping\_cart עם bookService.getAllProducts ו- cartService.viewCart בשביל לעדכן את סל הקניות שלנו. |
| error | עמוד תקלות כאשר משהו משתבש. |
| Index | עמוד הבית עבור guests. מראה את רשימת הספרים בלבד ומאפשר חיפוש ספרים.   * כאשר עושים חיפוש, אנחנו משתמשים בקוד javascript שמובנה בתוך העמוד, והוא מסננן לנו את התוצאות לפי מה שהוזן. * בעמוד מספר קישורים שמעבירים אותנו או לעמוד login או לעמוד register. |
| login | עמוד ההתחברות למשתמש עבור guests.   * הזנת username ו- password מטופלים ע"י spring security. * במידה והוכנסו פרטים שגויים, מטפל בזה loginError(Model model) * במידה והתחברנו בהצלחה, מועברים לעמוד ב-dashboard המתאים. |
| order\_customer | עמוד של admin. צפייה בפרטי לקוח עבור הזמנה מסוימת.   * הפרטים שמוראים מוזנים כבר מה-controller. |
| order\_products | עמוד של או admin או member. מאפשר צפייה ברשימת ספרים עבור הזמנה.   * הפרטים של הספרים מוזנים משכבת ה-controller. |
| orders | עמוד של או admin או member. אם member, נותן רשימת כל ההזמנות שעשה. אם admin, נותן רשימה כללית של כל ההזמנות.   * כאשר לוחצים על כפתור view customer , מופעל getCustomerByOrder ב- orderController. הוא מזמן את getCustomerFromAnOrder ב- orderService, שמחזיר את פרטי הלקוח, ואז אנחנו מועברים לעמוד order\_customer עם הנתונים. * כאשר לוחצים על כפתור view Ordered Books , מופעל getById ב- orderController. הוא מזמן את getCustomerFromAnOrder ב- orderService, שמחזיר את פרטי הלקוח, ואז אנחנו מועברים לעמוד order\_customer עם הנתונים. |
| register | עמוד ההרשמה של משתמש member חדש.   * בלחיצה על כפתור Register, מופעל registerForSubmission מ- AuthenticationController, שקודם מזמן את userService.getUserByUsername כדי לנסות לשלוף משתמש. אם המשתמש קיים, אז נותן הודעה מתאימה. אם לא קיים, אז יופעל userService.save שישמור את המשתמש החדש. * כל השדות צריכים להיות תקינים, אחרת תוחזר הודעה מתאימה. |
| shopping\_cart | עמוד שבוא אנחנו מסיימים את ההזמנה שלנו.   * אנחנו נראה את פרטי ההזמנה ויהיה אפשר או למחוק פריט מהזמנה או לסיים את ההזמנה. ניתן גם לחזור לעמוד הבית כדי להוסיף עוד פריט להזמנה. * בלחיצה על Remove Book, יופעל removeFromCart, שבהתאם מפעיל את cartService.removeFromCart, פעולה שמסירה את הפריט מסל הקניות. * בלחיצה על checkout, מופעל checkout ב- cartController. קודם כל, מופעל cartService.viewCart כדי לבדוק האם הסל קניות ריק. אם לא, מופעל cartService.checkout, שהיא מתודה מורכבת אשר שולפת את המשתמש בעזרת userRepository.findUserByUsername. אם המשתמש קיים, אז עושים סכום עבור כל הספרים, ואז מפעילים את bookRepository.save כדי להוריד את כמות הספרים מהמלאי. אחרי שחיסרנו את כל הספרים מהמלאי וספרנו את הסכום, נשמור את ההזמנה בעזרת orderRepository.save, נקשר את ההזמנה למשתמש בעזרת userRepository.save, וננקה את סל הקניות, לפני שמועברים לדף checkout. |
| single\_product | מראה פרטים עבור ספר אחד מתוך הזמנה.   * פרטי הספר מוזנים בשכבת ה-controller. |
| update\_product | מעדכן את המידע של ספרים.   * בלחיצה על כפתור submit, מופעלת פעולה ב-BookController מסוג post, updateProduct, שמתחברת לפעולה saveProduct ב- bookService ששומרת את השינויים שעשינו ב-database. |
| upload\_product | עמוד להוספת ספר חדש.   * בלחיצה על כפתור submit, קודם נבדק האם הקלט תקין. אם לא, תוחזר הודעה מתאימה. * אם הקלט תקין, אז יופעל saveUser מ- bookController, ששומר את ה- book שלנו. המתודה תפעיל את bookService.saveProduct, שמפעילה את bookRepository.save, ששומרת את הספר החדש שלנו ב-database. |

הערה: קבצי CSS לא מוצגים כי הם רק לעיצוב.

חבילת Controller:

חבילה זאת מכילה מספר קבצי java שמטפלים באינטראקציה בין קבצי ה-html לקוד ה-java. הם מגדירים את ה-url ומפעילים קוד על הדפים כאשר פעולות מסוימות מתקיימות.

מחלקת AdminController:

המחלקה השולטת עבור פעולות admin.

|  |  |
| --- | --- |
| מתודות | הסבר |
| String productsList(Model model) | נותן את רשימת הספרים עבור admin\_dashboard |
| String getAllOrders(Model model) | מסיג את כל ההזמנות שלקוחות בצעו עבור orders. |

מחלקת AuthenticationController:

המחלקה הזאת נועדה בשביל לבדוק התחברות תקינה של משתמשים.

|  |  |
| --- | --- |
| מתודות | הסבר |
| String login() | מתודה זו מטפלת בבקשה של משתמש להיכנס לעמוד התחברות. Springsecurity מטפל בשאר. |
| String loginError(Model model) | אם למשתמש אין הרשאות מתאימות, מתודה זאת מטפלת בבעיה. |
| String register(Model model) | מתודה זו מטפלת בבקשה של משתמש להיכנס לעמוד יצירת משתמש |
| String registerForSubmission(@Valid User user, BindingResult result, Model model) | מתודה זאת מטפלת ביצירה של משתמש חדש ובודקת אם הקלט תקינה. |

מחלקת BookController:

המחלקה השולטת על הספרים, כולל צפייה, יצירה, עריכה, ומחיקה

|  |  |
| --- | --- |
| מתודות | הסבר |
| String getProductUploadForm(Model model) | מתודה שמעבירה אותנו לעמוד הוספת ספר חדש |
| String saveUser(Book book) | מתודה להוספת/עריכת ספרים במסד הנתונים |
| String productsList(Model model) | מתודה לצפייה בכל הספרים במסד הנתונים |
| String getProductById(Model model, @PathVariable(name = "id") Long productId) | מתודה לצפייה בפרטים של ספר יחיד |
| String deleteProductById(@PathVariable(name = "id") Long productId) | מתודה זאת מוחקת ספר מה-front אבל משאירה אותו במסד הנתונים |
| String updateProduct(Model model, @PathVariable(name = "id") Long productId) | המתודה הזאת מעבירה אותנו לעמוד עדכון ספרים. |
| String updateProduct(@PathVariable(name = "id") Long productId, Book book) | המתודה הזאת מעדכנת את הספר ומחזירה אותנו לעמוד הראשי של ה-admin |

מחלקת CartController:

המחלקה השולטת על סל הקניות

|  |  |
| --- | --- |
| מתודות | הסבר |
| String addToCart(@PathVariable(name = "id") Long productId, @RequestParam(name = "quantity") int quantity, Model model) | המתודה הזאת נועדה כדי להוסיף ספרים לסל הקניות. היא מקבלת id של הספר שאנחנו רוצים להוסיף. |
| String viewCart(Model model) | המתודה הזאת נועדה עבור צפייה בסל הקניות. היא תראה את כל הספרים שהוספנו לסל. |
| String checkout(Model model) | המתודה הזאת מופעלת כשאר לוחצים על סיום ההזמנה, ומוסיפה את ההזמנה למסד הנתונים אם לא ריקה. |

מחלקת HomeController:

המחלקה השולטת על עמוד הבית

|  |  |
| --- | --- |
| מתודות | הסבר |
| String home(Model model) | מחלקה זאת נועדה כדי להבדיל אם המשתמש מסוג admin או מסוג member ומעביר אותנו לעמוד מתאים. אם מסוג guest, אז יראה רק את רשימת הספרים. |

מחלקת HomeController:

המחלקה השולטת על עמוד הבית

|  |  |
| --- | --- |
| מתודות | הסבר |
| String home(Model model) | מחלקה זאת נועדה כדי להבדיל אם המשתמש מסוג admin או מסוג member ומעביר אותנו לdashboard מתאים. אם מסוג guest, אז יעביר אותנו ל-index. |

מחלקת OrderController:

המחלקה השולטת על ההזמנות.

|  |  |
| --- | --- |
| מתודות | הסבר |
| String getProductsByOrderId(@PathVariable(name = "id") Long orderId, Model model) | מתודה שמראה את פרטי המוצרים בהזמנה |
| String getCustomerByOrder(@PathVariable(name = "id") Long orderId, Model model) | מתודה שמראה את פרטי הלקוח של הזמנה |

מחלקת UserController:

המחלקה השולטת על המשתמשים.

|  |  |
| --- | --- |
| מתודות | הסבר |
| String orders(Model model) | מתודה שמראה את כל ההזמנות שלקוח מחובר עשה, ולא תראה הזמנות של לקוחות אחרים. |

חבילת Model:

החבילה הזאת מכילה את כל הטבלאות במסד הנתונים. (מופיע ב-ERD)

חבילת Repository:

חבילה שמכילה שאילתות עבור מסד הנתונים.

חבילת security:

החבילה בשביל הגדרות האבטחה של האתר.

מחלקת WebSecurityConfig:

המחלקה הזאת מכילה את כל מתודות האבטחה וההגדרות של האתר. TODO

|  |  |
| --- | --- |
| מתודות | הסבר |
| void configure(AuthenticationManagerBuilder auth) | מתודה שמופעלת כאשר המשתמש מתחבר ל-user שלו. |
| void configure(HttpSecurity http) | מתודה שבודקת אילו בקשות משתמש ניתן לאשר ואילו בקשות משתמש לא ניתן לאשר. היא בודקת איזה משתמש יכול לקבל גישה ל-path או פעולות מסויימות. |
| BCryptPasswordEncoder encoder() | מתודת bean עבור קידוד סיסמא בעת התחברות למשתמש.(בשימוש ב-userService) |
| void configure(WebSecurity web) | מתודה בשביל הגדרת url path עבור דפי ה-html. |

חבילת service:

החבילה בשביל מתודות שירות שמטפלות בפעולות שירות מסובכות יותר, כולל גישה למסד הנתונים ופונקציות אחרות.

מחלקת BookService:

המחלקה הזאת מכילה מתודות שירות עבור books.

|  |  |
| --- | --- |
| מתודות | הסבר |
| void saveProduct(Book book) | מתודה עבור שמירת ספר במסד הנתונים |
| List<Book> getAllProducts() | מתודה שמסיגה את כל הספרים ממסד הנתונים |
| List<Book> getAllUnDeletedProducts() | מתודה שמסיגה את כל הספרים, כולל ספרים שעדיין לא נמחקו |
| Book getProductById(Long productId) | מתודה שמסיגה ספר על פי ה-ID שלו. |
| void deleteProduct(Long productId) | מתודה עבור מחיקה של ספר לפי ה-ID. ספר מחוק לא באמת מוסר ממסד הנתונים, אלה מסומן כמחוק. |

מחלקת CartService:

המחלקה הזאת מכילה מתודות שירות עבור ה-cart

|  |  |
| --- | --- |
| מתודות | הסבר |
| removeFromCart(Long productId) | מתודה להסרת ספר מסל הקניות |
| void addToCart(Long productId, int quantity) | מתודה להוספת ספר וכמות לסל הקניות |
| List<Book> viewCart() | מתודה לצפייה בכל המוצרים בסל הקניות |
| Order checkout() | מתודה שמבצעת את לוגיקת ה- checkout בסיום הזמנה. |

מחלקת OrderService:

המחלקה הזאת מכילה מתודות שירות עבור orders.

|  |  |
| --- | --- |
| מתודות | הסבר |
| List<Order> getAllOrders() | מתודה ששולפת את כל ההזמנות ממסד הנתונים |
| List<Order> getOrdersByUser(String username) | מתודה ששולפת את כל ההזמנות של משתמש מסוים |
| List<OrderItem> getBooksByOrderId(Long orderId) | מתודה ששולפת את הספרים מתוך הזמנה |
| User getCustomerFromAnOrder(Long orderId) | מתודה ששולפת את הלקוח מתוך הזמנה |

מחלקת UserService:

המחלקה הזאת מכילה מתודות שירות עבור Users.

|  |  |
| --- | --- |
| מתודות | הסבר |
| Optional<User> getUserByUsername(String username) | מתודה שמחזירה משתמש לפי שם משתמש |
| User save(User user) | מתודה לשמירה של משתמש בתוך מסד הנתונים |
| UserDetails loadUserByUsername(String username) | מתודת מפתח עבור spring security. Spring security משתמשת במתודה זאת כדי למצוא משתמש במסד הנתונים ולאשר אותו. |
| Set<SimpleGrantedAuthority> getAuthority(User user) | מתודה שמסיגה את ה- role של משתמש ממסד הנתונים. (כלומר או member או admin) |

קבצים חשובים נוספים:

הינה קבצים והגדרות נוספות שלא בחבילה ספציפית

Application properties:

מכיל הגדרות עבור ה-h2 database.

BookOrderingSystem:

מכיל את מתודת main. במתודת ה-main גם מוגדרים מספר דברים שמוכנסים למסד הנתונים.

Pom.xml:

מכיל את ה-spring dependencies של הפרוייקט.