**Diagramy  
(**[**https://www.planttext.com/**](https://www.planttext.com/)**)**

1. Diagram aktywności rejestracja użytkownika

@startuml

|User Interface|

start

:User opens registration page;

:User fills out registration form;

:Submit registration data;

|System|

:Receive and validate input;

if (Is the data valid?) then (No)

:Return an error;

:Display error message to user;

stop

else (Yes)

:Check if user already exists;

|Database|

:Search for existing user record;

:Send query result;

|System|

if (User exists?) then (Yes)

:Return an error "User already exists";

:Display error message to user;

stop

else (No)

:Encrypt password;

:Create new user record;

|Database|

:Store user in database;

|System|

:Generate success response;

|User Interface|

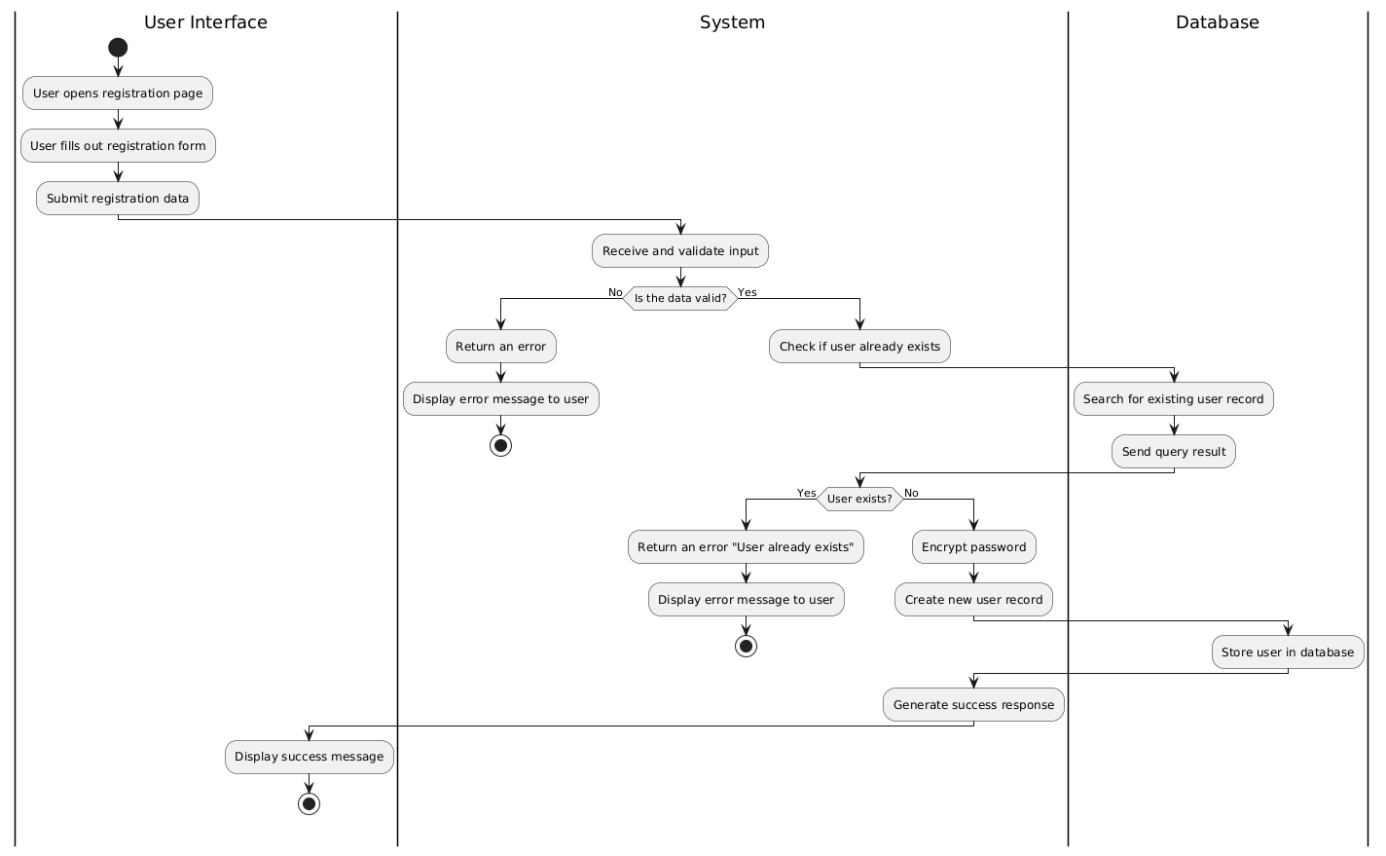
:Display success message;

stop

endif

endif

@enduml



2. Diagram sekwencji rejestracja użytkownika

@startuml

title Sequence Diagram: Register User

actor User as U

participant "Registration Form" as RF

participant System

database "Database" as DB

== Main Flow ==

U -> RF: openRegistrationPage()

RF -> U: displayRegistrationForm()

U -> RF: submitRegistrationData(username, password, email)

RF -> System: processRegistrationData(username, password, email)

System -> DB: checkIfUserExists(username)

DB --> System: userExistsResult(no)

System -> System: encryptPassword()

System -> DB: createUserRecord(encryptedPassword, email)

DB --> System: userRecordCreated(success)

System --> RF: returnSuccessMessage()

RF --> U: displaySuccessMessage()

== Alternate Flow ==

alt User Already Exists

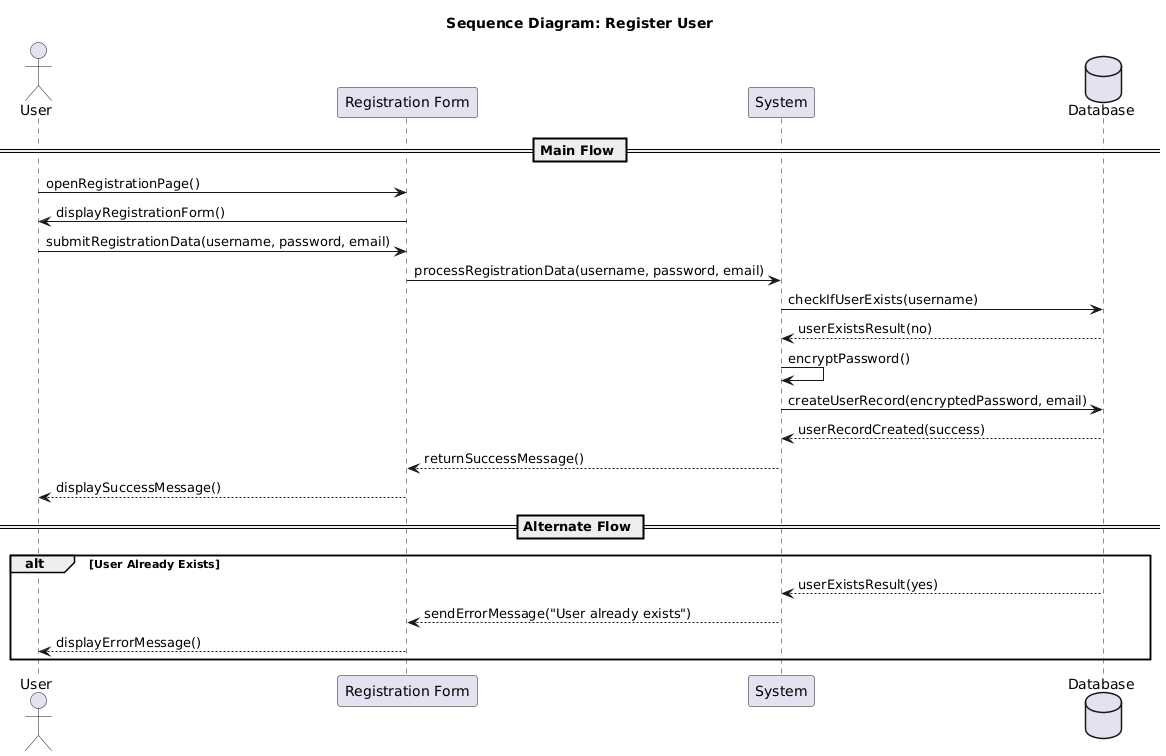
DB --> System: userExistsResult(yes)

System --> RF: sendErrorMessage("User already exists")

RF --> U: displayErrorMessage()

end

@enduml



3. Diagram aktywności dodawanie książki

@startuml

|User Interface|

start

:User opens book management page;

:User fills out book details;

:Submit book data;

|System|

:Receive and validate input;

if (Is the data valid?) then (No)

:Return an error;

:Display error message to user;

stop

else (Yes)

:Check if book already exists;

|Database|

:Search for book record;

:Send query result;

|System|

if (Book exists?) then (Yes)

:Return an error "Book already exists";

:Display error message to user;

stop

else (No)

:Create book record;

|Database|

:Store book in database;

|System|

:Generate success response;

|User Interface|

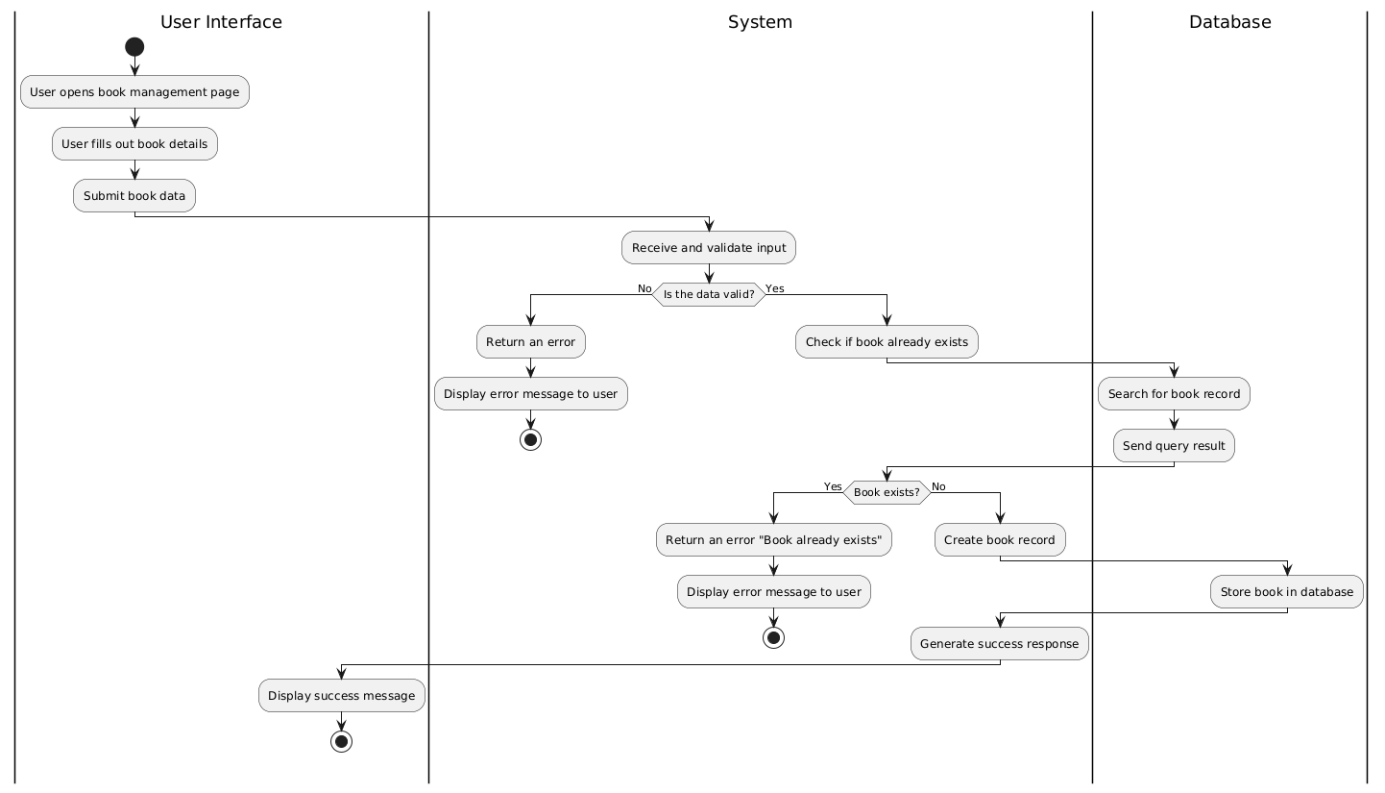
:Display success message;

stop

endif

endif

@enduml



4. Diagram sekwencji dodawanie książki

@startuml

title Sequence Diagram: Add Book

actor User as U

participant "Book Form" as BF

participant System

database "Database" as DB

== Main Flow ==

U -> BF: openAddBookPage()

BF -> U: displayBookForm()

U -> BF: submitBookForm(title, author, details)

BF -> System: processBookData(title, author, details)

System -> DB: checkIfBookExists(title, author)

DB --> System: bookExistsResult(no)

System -> System: createNewBookRecord()

System -> DB: saveBook(title, author, details)

DB --> System: bookSaved(success)

System --> BF: returnSuccessMessage()

BF --> U: displaySuccessMessage()

== Alternate Flow ==

alt Book Already Exists

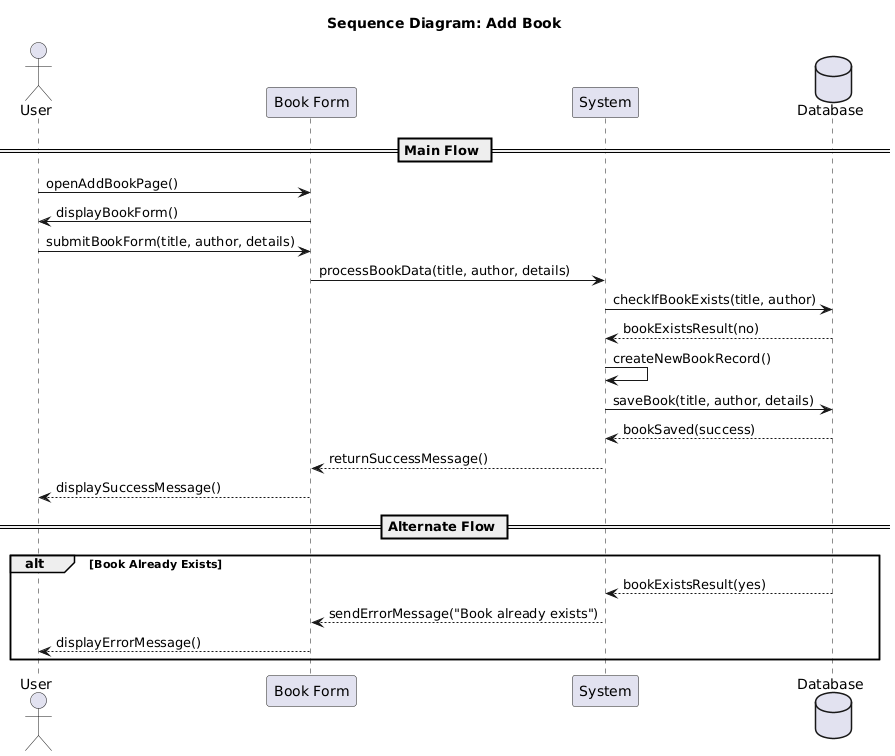
DB --> System: bookExistsResult(yes)

System --> BF: sendErrorMessage("Book already exists")

BF --> U: displayErrorMessage()

end

@enduml



5. Diagram aktywności rezerwacja książki

@startuml

|User Interface|

start

:User opens book management page;

:User fills out book details;

:Submit book data;

|System|

:Receive and validate input;

if (Is the data valid?) then (No)

:Return an error;

:Display error message to user;

stop

else (Yes)

:Create book record;

|Database|

:Store book in database;

|System|

:Generate success response;

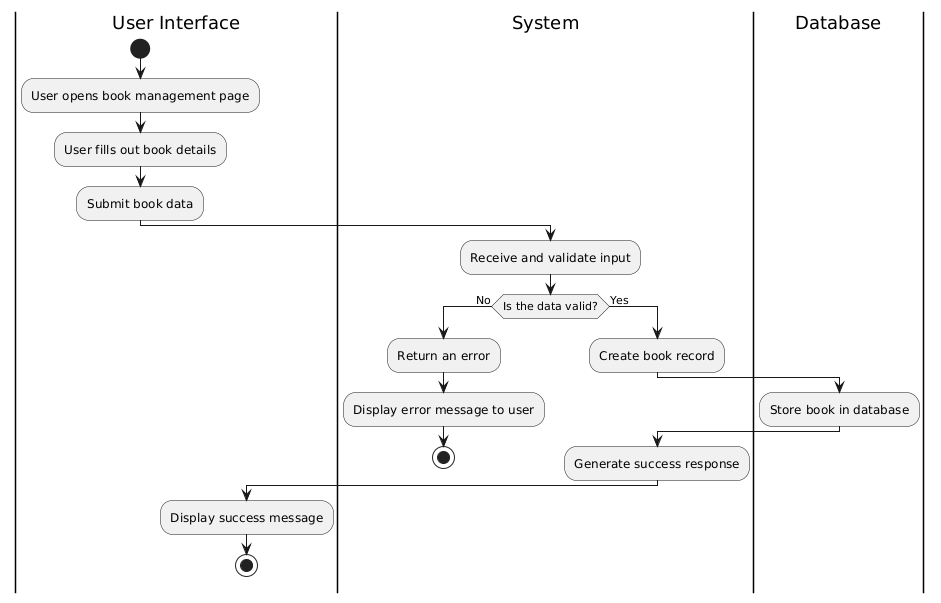
|User Interface|

:Display success message;

stop

endif

@enduml



6. Diagram sekwencji rezerwacja książki

@startuml

title Sequence Diagram: Add Book

actor User as U

participant "Book Form" as BF

participant System

database "Database" as DB

== Main Flow ==

U -> BF: openAddBookPage()

BF -> U: displayBookForm()

U -> BF: submitBookForm(title, author, details)

BF -> System: processBookData(title, author, details)

System -> System: validateBookData()

alt Data Invalid

System --> BF: sendErrorMessage("Invalid data")

BF --> U: displayErrorMessage()

else Data Valid

System -> DB: createNewBookRecord(title, author, details)

DB --> System: bookCreated(success)

System --> BF: returnSuccessMessage()

BF --> U: displaySuccessMessage()

end

@enduml

