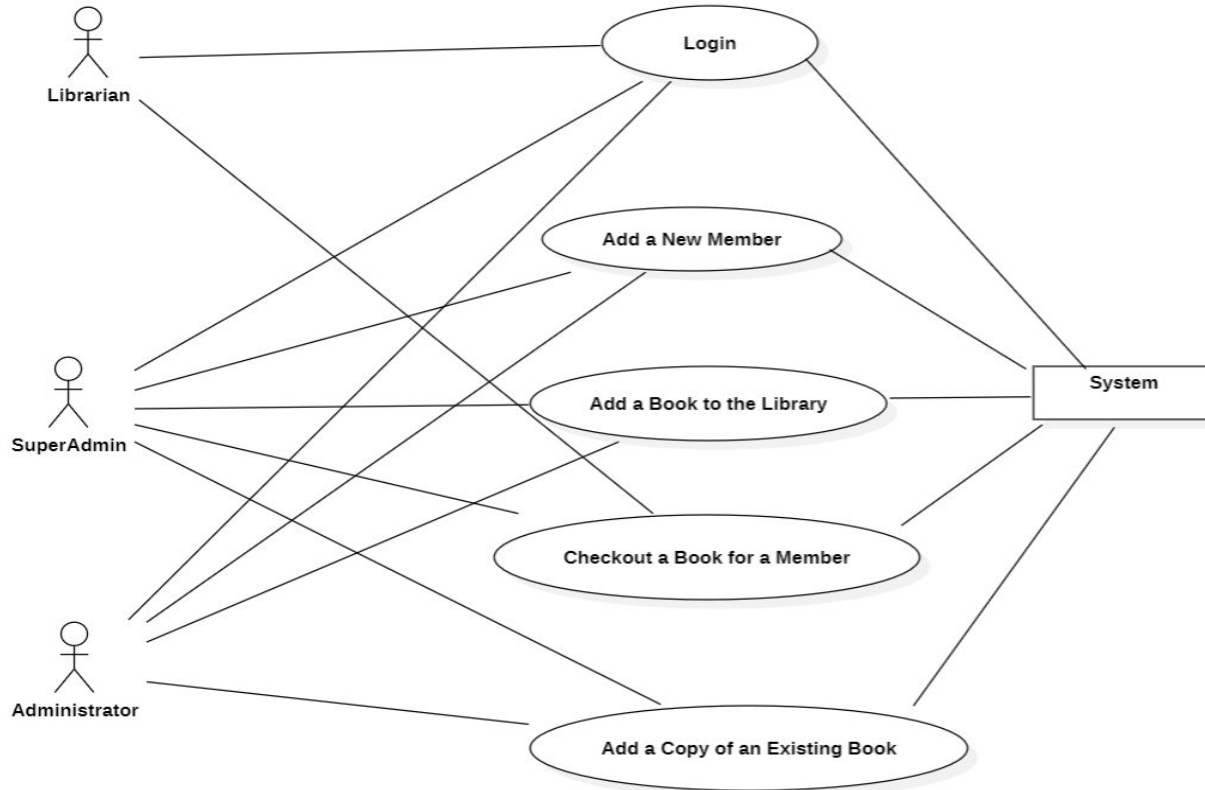# Library System
Group 4

Members:
Awais Waheed - 618677
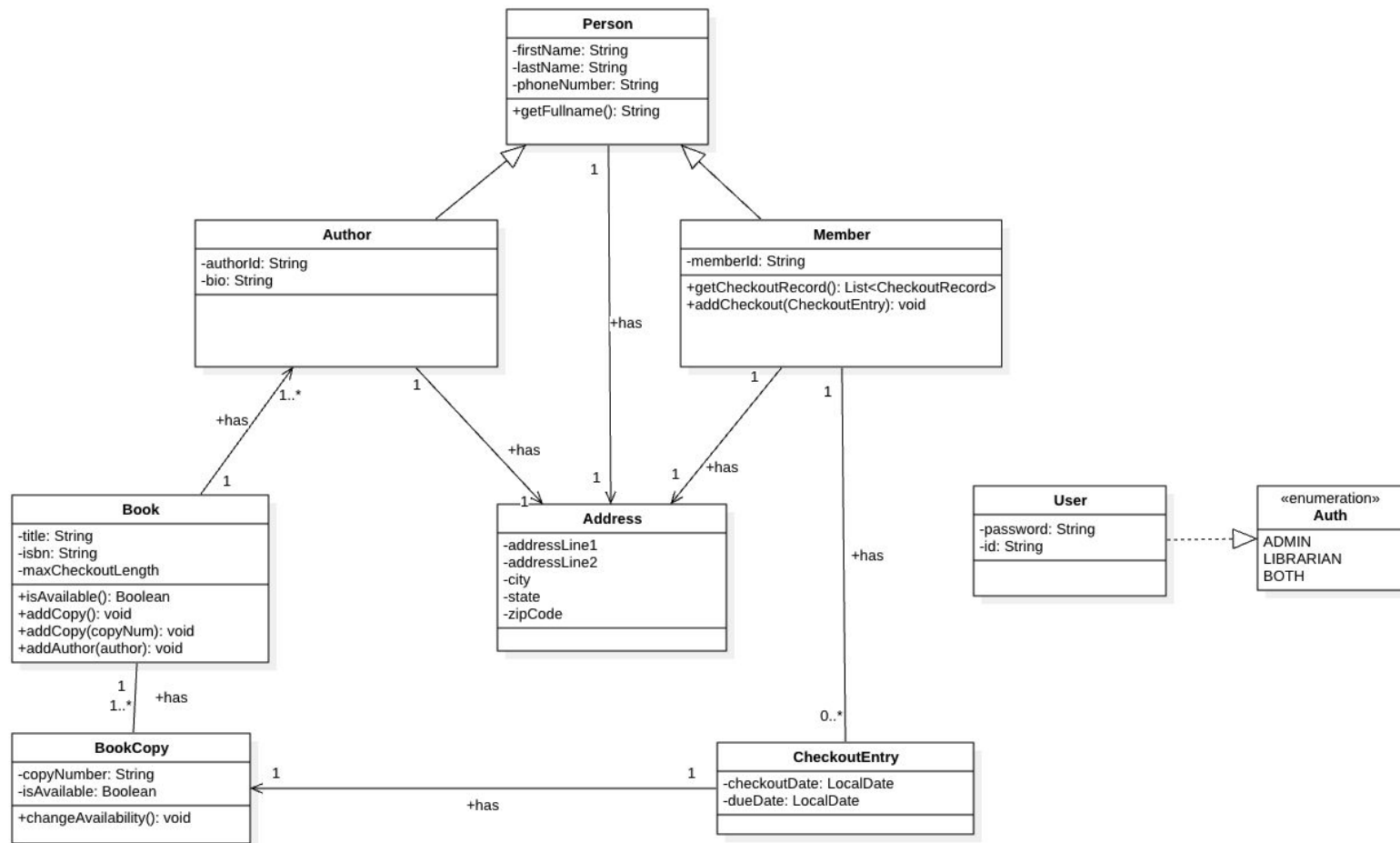Arslanbek Ametov - 618653

Use cases

- Login
- Add a new library member
- Add a new book
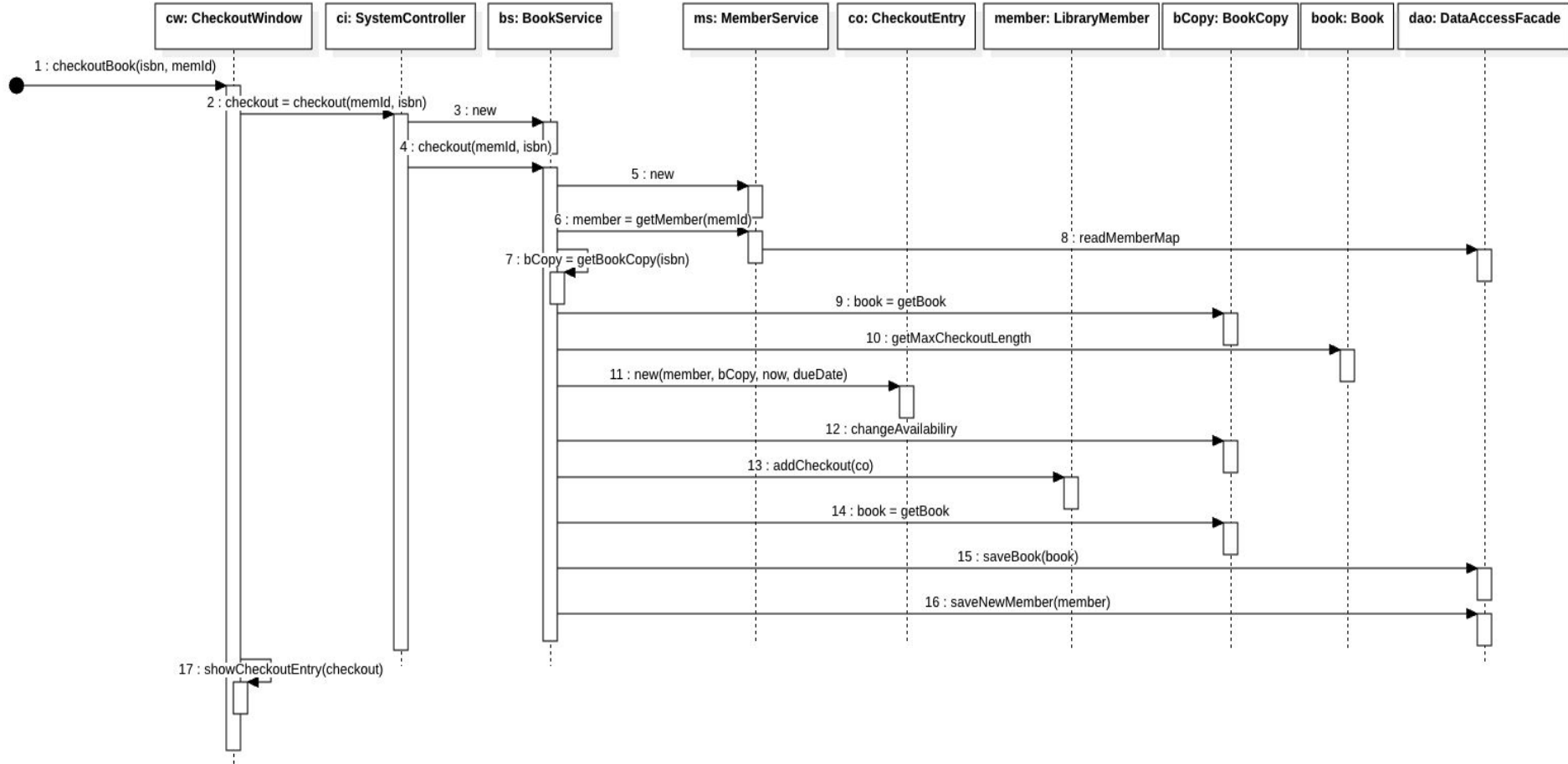- Checkout a book
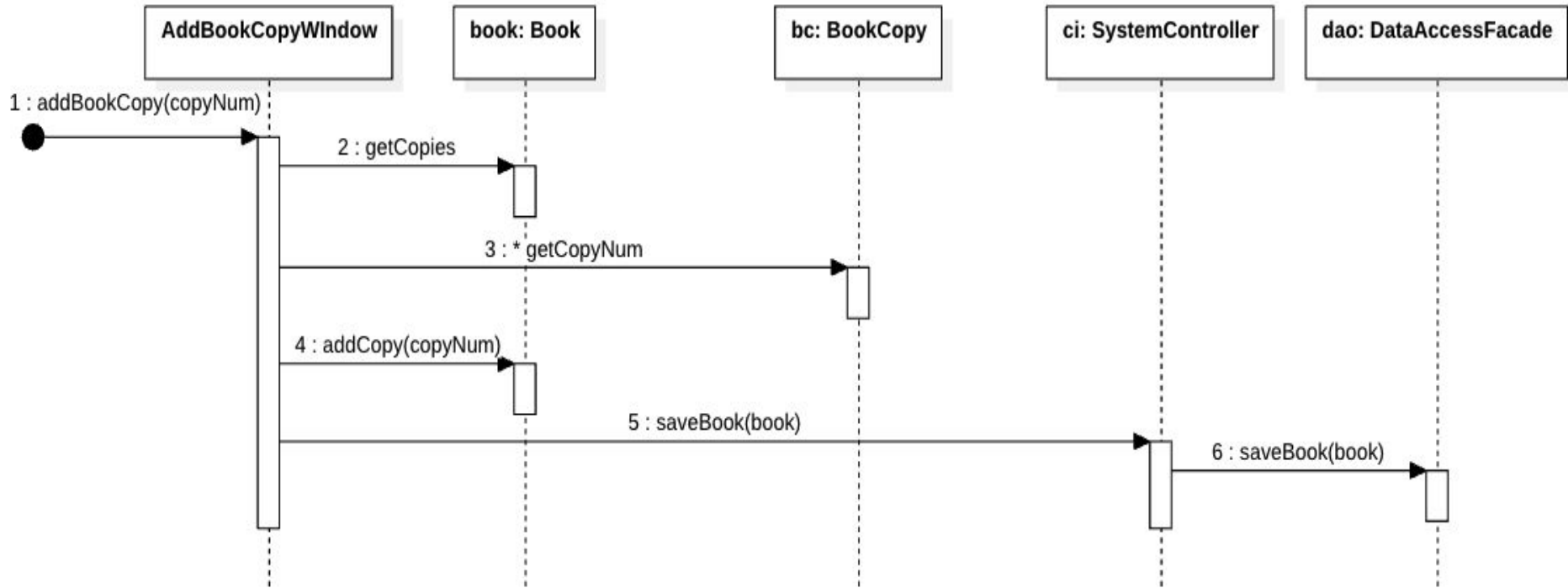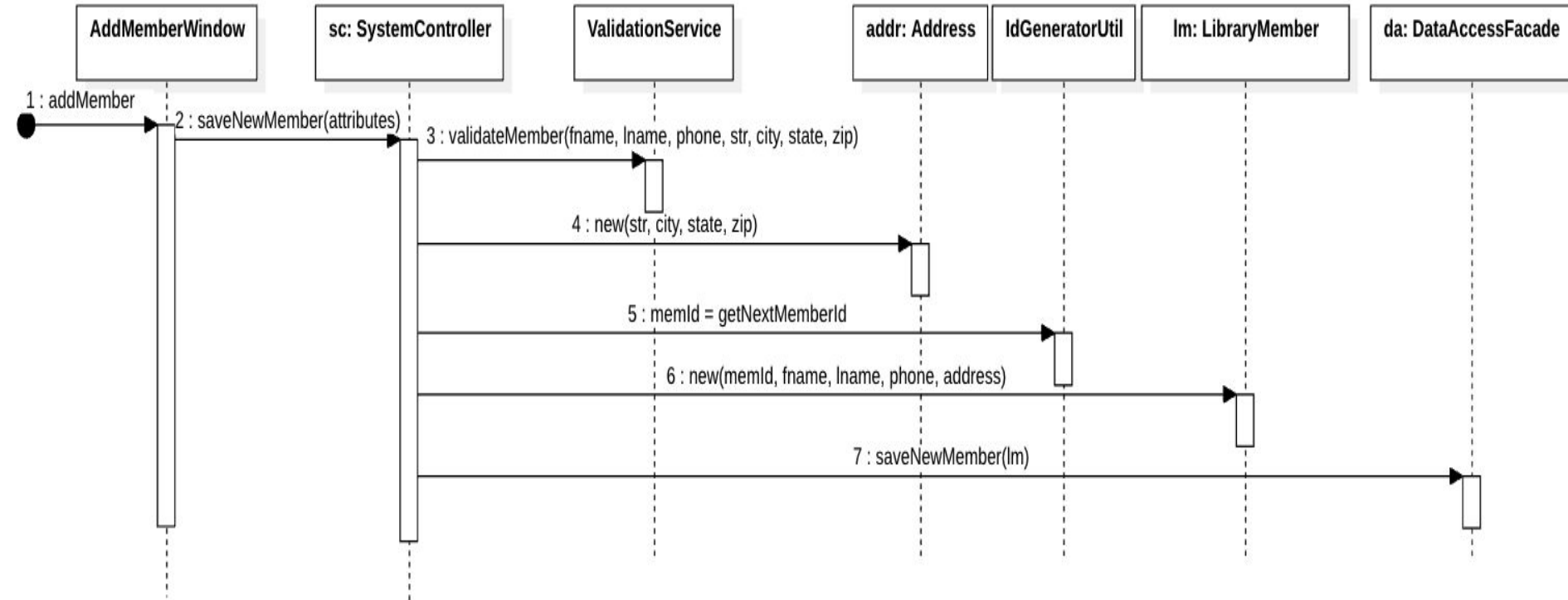- Add a copy of an existing book

Use-case diagram

Class diagram

Sequence diagram: Checkout book

Sequence diagram: Add a copy of an existing book

Sequence diagram: Add a new member

Final Class Model: Add new member

AddMemberWindow

+AddMember()

+calls saveNewMember →

«interface»
ControllerInterface

+saveNewMember(fname, lname, phone, state, str, city, zip)

ValidationService

+ValidateMemberInput(fname, lname, phone, state, str, city, zip)

+implements

+asks to validate

SystemController

+saveNewMember(fname, lname, phone, state, str, city, zip)

+ask to generate unique id

IdGeneratorUtil

+GetNextMemberID(): String

+persists member in memory

+creates

+uses

DataAccessFacade

+saveNewMember(mem)

LibraryMember

Address

Validation rules discussion

In any application, especially data-driven systems like a library management system, data validation plays a critical role in ensuring both data integrity and system stability. Without proper validation, systems can become prone to errors, security risks, and inconsistent data.

1. **Data Integrity.** The foremost goal of validation is to ensure that the data entered into the system is accurate, consistent, and follows the expected format. In the example code, we see validation checks for ISBN format, phone numbers, zip codes, and states. These ensure that only correctly formatted data is processed.

2. **Error Prevention**: Invalid data can lead to malfunctioning applications or corrupt databases. Validation rules prevent this by catching errors before they are entered into the system. For example, in our code, if the ISBN format is incorrect or the number of book copies is not greater than 0, the system stops the process and informs the user to correct the input.

Validation rules discussion

3. **User Experience**: Good validation improves the user experience by providing immediate feedback. Clear, actionable error messages guide users on how to correct their input. In our implementation, the messages such as "Invalid ISBN format" or "Please select at least one author" are straightforward and help the user understand and fix mistakes quickly.

4. **Security**: Input validation is a critical security practice. Proper validation prevents harmful input from entering the system, which could lead to vulnerabilities like SQL injection, buffer overflows, or other security issues. For instance, validating inputs such as telephone numbers and addresses ensures that no harmful data can be injected into the system.

5. **Separation of Concerns**: The code we've written follows a good separation of concerns. Validation logic is centralized in the `ValidationService` class, making the codebase cleaner and easier to maintain. Centralizing validation logic also allows for easy modifications or updates in the future.

Examples of Validation

1. **ISBN Validation:** The system requires ISBNs to follow a strict format (`\\d{2}-\\d{5}`). This prevents malformed ISBNs from being added to the system, ensuring the integrity of book records.

2. **Author Selection**: Validating that at least one author is selected ensures completeness of book information and avoids scenarios where a book might be entered without proper attribution.

3. **State Code Validation**: The code checks that the state is a valid two-letter state code. This maintains consistency across geographical data and prevents errors caused by incorrect state information.

4. **Field Requirements**: Essential fields like first name, last name, street, city, and zip code are marked as mandatory, ensuring that important user details are always provided.

SCI knowledge: Ensuring Data Integrity Through Validation Rules

In the project, validation rules ensure that only accurate and valid data enters the system, much like how SCI emphasizes maintaining purity and integrity in our actions and thoughts to uphold natural order. Just as data validation prevents corrupt information from destabilizing a system, SCI teaches that aligning ourselves with natural law leads to stability, harmony, and efficiency in life. Following the natural laws of consciousness allows for the free flow of creative intelligence, just as validation rules ensure the smooth functioning of an application by blocking errors.

SCI knowledge: Enhancing User Experience Through Simplicity

One of the main focuses in the project is improving user experience by providing clear, actionable feedback through validation. This reflects SCI's principle of simplicity. When the mind is in a state of coherence and simplicity, solutions come naturally and effortlessly. In software development, simplifying the user experience through intuitive validation mirrors how SCI teaches that when life aligns with the simple laws of nature, complexity dissolves, and efficiency emerges.

Demo

Questions?

Thank you!