

# VeriGen - NLP Requirements Analysis System

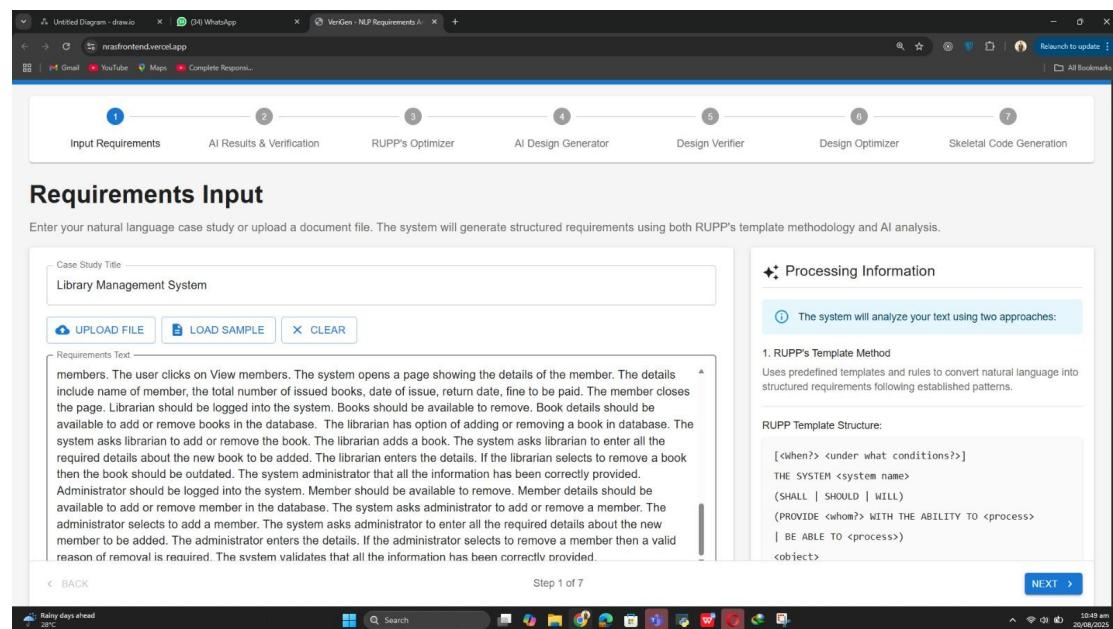
## Quick User Guide

### STEP 1: Input Requirements

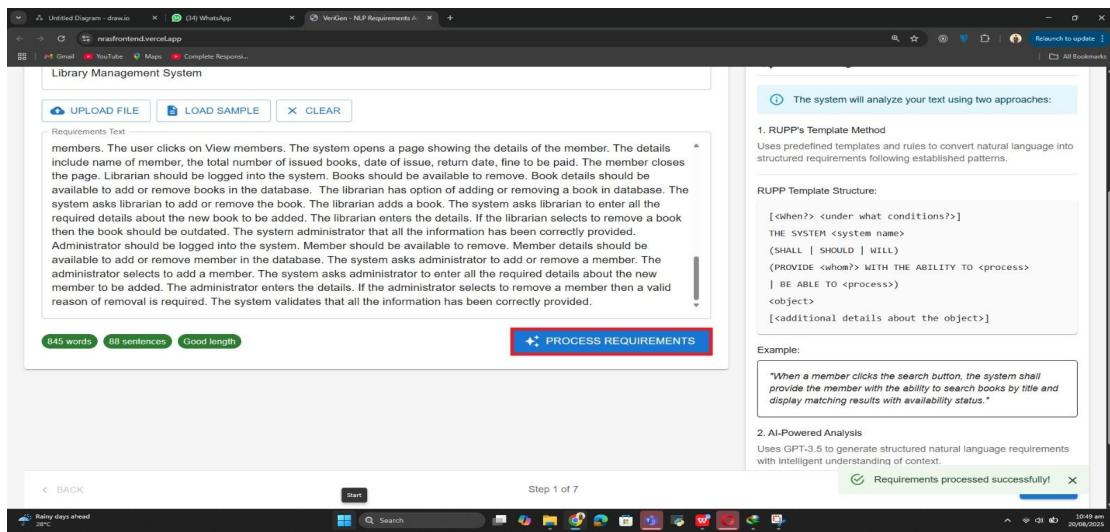
1. Enter your **Case Study Title** in the text field (e.g., "Library Management System")
2. Type or paste your requirements description in the **Requirements Text** area
3. Alternatively, click [**UPLOAD FILE**] to upload a document or [**LOAD SAMPLE**] to load an example
4. Click the [**PROCESS REQUIREMENTS**] button

**Result:** System will analyze your text and show "**✓ Requirements processed successfully!**"

- Shows the Requirements Input screen with text area and process button.



- Shows the processing complete with word count statistics and success message

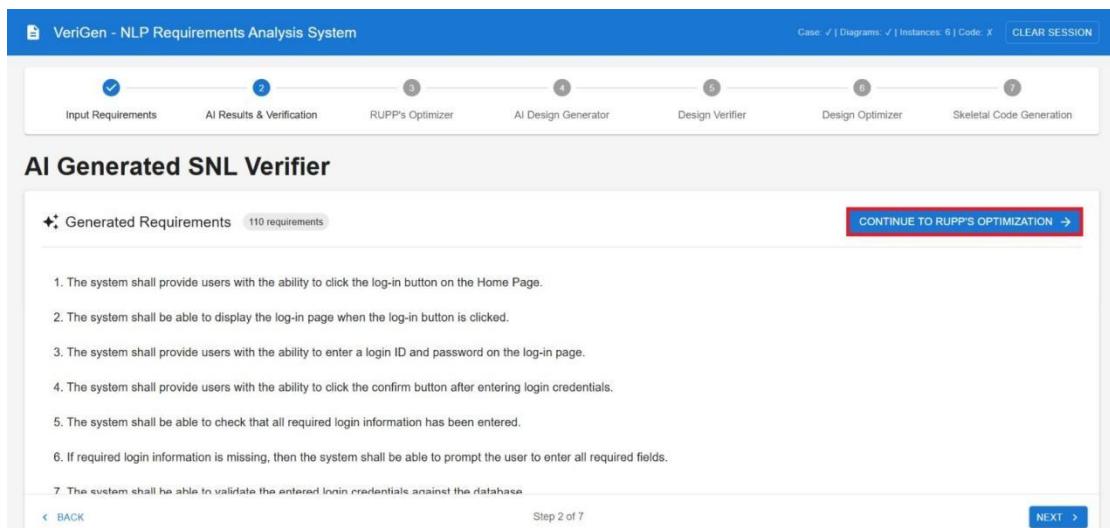


## STEP 2: AI Results & Verification

1. Review the AI-generated structured requirements list (110 requirements)
2. Click the [CONTINUE TO RUPP'S OPTIMIZATION →] button

**Result:** System displays AI Generated SNL Verifier with categorized requirements (Correct, Incorrect, Overspecified, Missing)

- Shows AI Generated SNL Verifier with requirements categories



- Shows detailed view of AI SNL Verifier categories (Correct in AI, Incorrect in AI, Overspecified, Missing)

The screenshot shows a dashboard titled "AI SNL Verifier" with four main sections:

- Correct in AI**: Requirements where AI correctly matched RUPP specifications. Examples include:
  - 11. The system shall provide members with the ability to close the member details page.
  - 12. The system shall provide librarians with the ability to add a book to the database.
  - 13. If the administrator selects to remove a member, then the system shall require a valid reason for removal.
- Incorrect in AI**: Requirements where AI made factual errors or misinterpretations. Examples include:
  - 1. If the librarian selects to add a book, then the system shall prompt the librarian to enter all required details about the new book.
  - 2. The system shall require members to provide their member ID to the librarian for book return.
  - 3. The system shall verify that book details are available before allowing addition or removal in the database.
  - 4. If required login information is
- Overspecified in AI**: Requirements where AI was too detailed beyond RUPP scope. Examples include:
  - 14. If a user attempts to reserve a book, then the system shall verify that a correct Book ID has been provided.
  - 15. If a user attempts to reserve a book, then the system shall verify that the book is available for reservation.
  - 16. The system shall prompt users to log in before allowing book reservation.
- Missing in AI**: Requirements that AI failed to capture. Examples include:
  - 3. 76. The system shall provide librarian with the ability to enter the details.
  - 4. 40. If the selected book is already reserved on another id then the system shall be able to d asks user to select another book.
  - 5. 10. The system shall be able to open a page showing the details of the member.

## STEP 3: RUPP's Optimizer

1. Click the [⚡ OPTIMIZE REQUIREMENTS] button
2. Wait for optimization to complete
3. Click the [NEXT →] button

**Result:** System shows "✓ Optimization complete!" with 86 optimized requirements in RUPP template format

- Shows RUPP's Optimizer ready state with Optimize Requirements button

The screenshot shows the "VeriGen - NLP Requirements Analysis System" interface at Step 3 of 7. The top navigation bar includes links for Input Requirements, AI Results & Verification, RUPP's Optimizer, AI Design Generator, Design Verifier, Design Optimizer, and Skeletal Code Generation. The current step, "RUPP's Optimizer", is highlighted with a blue circle containing the number 3.

**AI Generated Rupp's Optimization**

**RUPP Template Processing**

**Ready for Optimization**

Click "Optimize Requirements" to process your requirements using the RUPP template methodology. This will generate structured, validated requirements based on industry standards.

**OPTIMIZE REQUIREMENTS**

Step 3 of 7

- Shows completed optimization with structured requirements list

The screenshot shows the VeriGen - NLP Requirements Analysis System interface. At the top, there is a navigation bar with tabs: Input Requirements, AI Results & Verification, RUPP's Optimizer (highlighted with a blue circle), AI Design Generator, Design Verifier, Design Optimizer, and Skeletal Code Generation. Below the navigation bar, the title "AI Generated Rupp's Optimization" is displayed. Underneath the title, there is a section titled "RUPP Template Processing" with a progress indicator showing "86 requirements processed". A large list of requirements follows, starting with: "1. The system shall provide member with the ability to click the login button on the home page.", and ending with "14. The system shall be able to stored books database.". At the bottom right of the main content area, a message box says "Optimization complete!". Navigation buttons "BACK" and "NEXT" are at the bottom left and right respectively.

## STEP 4: AI Design Generator

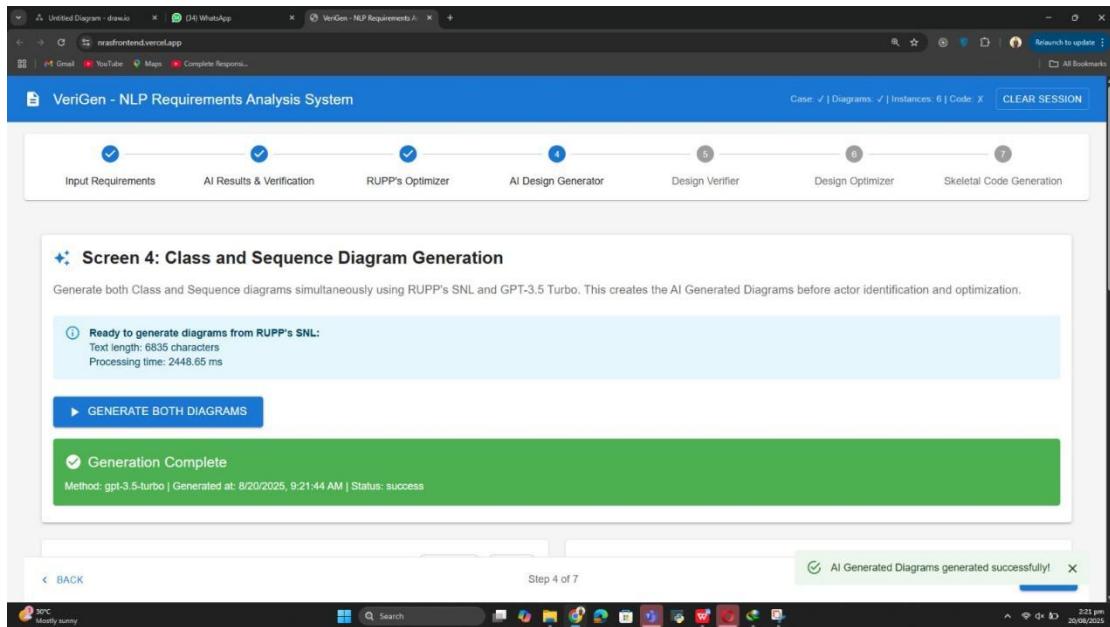
1. System displays "Screen 4: Class and Sequence Diagram Generation"
2. Click the [▶ GENERATE BOTH DIAGRAMS] button
3. Wait for generation to complete

**Result:** System shows "✓ AI Generated Diagrams generated successfully!" with Class Diagram (13 elements, 127 lines) and Sequence Diagram (16 elements, 141 lines)

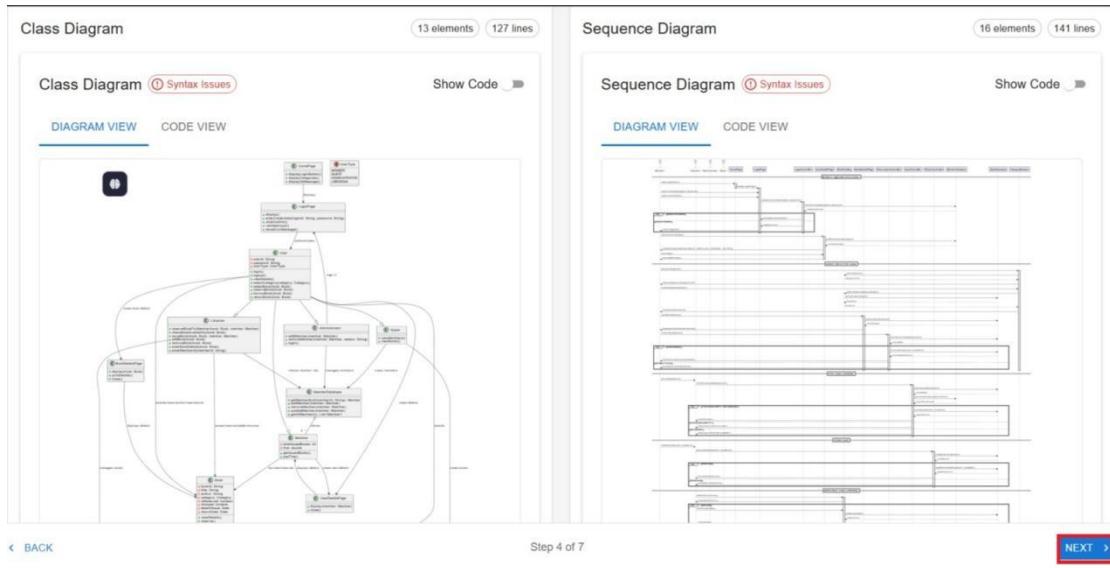
- Shows Screen 4 with Generate Both Diagrams button ready.

The screenshot shows the VeriGen - NLP Requirements Analysis System interface. The top navigation bar is visible with the "RUPP's Optimizer" tab highlighted. Below the navigation bar, the title "Screen 4: Class and Sequence Diagram Generation" is displayed. A message box contains the text: "Ready to generate diagrams from RUPP's SNL: Text length: 6838 characters Processing time: 2448.65 ms". Below this message is a prominent red button labeled "GENERATE BOTH DIAGRAMS". Navigation buttons "BACK" and "NEXT" are at the bottom left and right respectively. The status bar at the bottom right shows the date and time as "21.9 pm 20/09/2023".

- Shows generation complete with success message.



- Shows both generated diagrams side-by-side (Class Diagram on left, Sequence Diagram on right)



## STEP 5: Design Verifier

1. Click the [► IDENTIFY INSTANCES & VERIFY DIAGRAMS] button
2. Review the Verification Results showing:

- Verification Score
- Coverage Statistics
- Missing Instances
- Overspecified Classes (Librarian, Guest, Book, Administrator, Member)
- Incorrect Instances (HomePage, Category)
- Extra Instances (LoginPage, UserDetailsPage, MemberDatabase, BookDatabase, BookDetailsPage)
- Present Instances (User)

### 3. Click the [NEXT →] button

**Result:** System shows "✓ 6 actors identified and verified!" with detailed analysis

- Shows Design Verifier screen with Identify Instances button.

VeriGen - NLP Requirements Analysis System

Case: ✓ | Diagrams: ✓ | Instances: 6 | Code: X | CLEAR SESSION

Input Requirements   AI Results & Verification   RUPP's Optimizer   AI Design Generator   **5**   Design Verifier   Design Optimizer   Skeletal Code Generation

**Screen 5: Design Verifier**

Extract actors from original requirements using POS tagging and NER, then verify against the generated diagrams to identify missing or inconsistent elements.

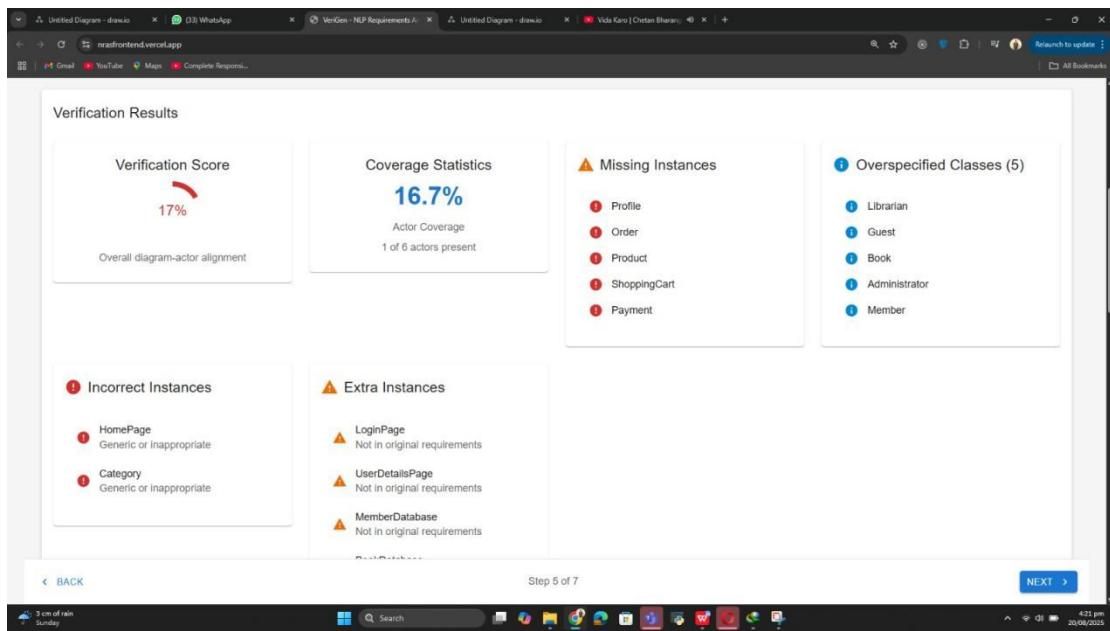
Analysis Method: Using spaCy NER + POS tagging + GPT-3.5 for comprehensive actor extraction.  
 Verification: Cross-referencing identified instances with diagram elements + overspecification detection.  
 Detection: Missing actors, overspecified classes, incorrect classes, and specification issues.  
 Status: ✓ Requirements available (4838 chars) ✓ Class diagram available (2880 chars) ✓ Sequence diagram available (5484 chars)

▶ IDENTIFY INSTANCES & VERIFY DIAGRAMS

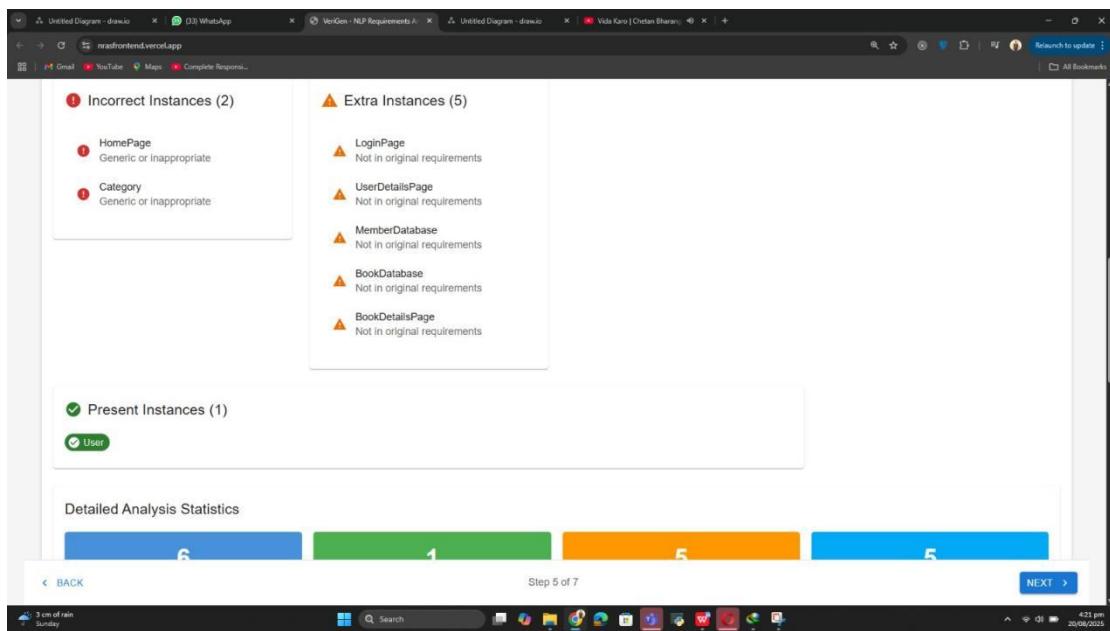
Verification Results

< BACK Step 5 of 7 ✓ 6 actors identified and verified! X

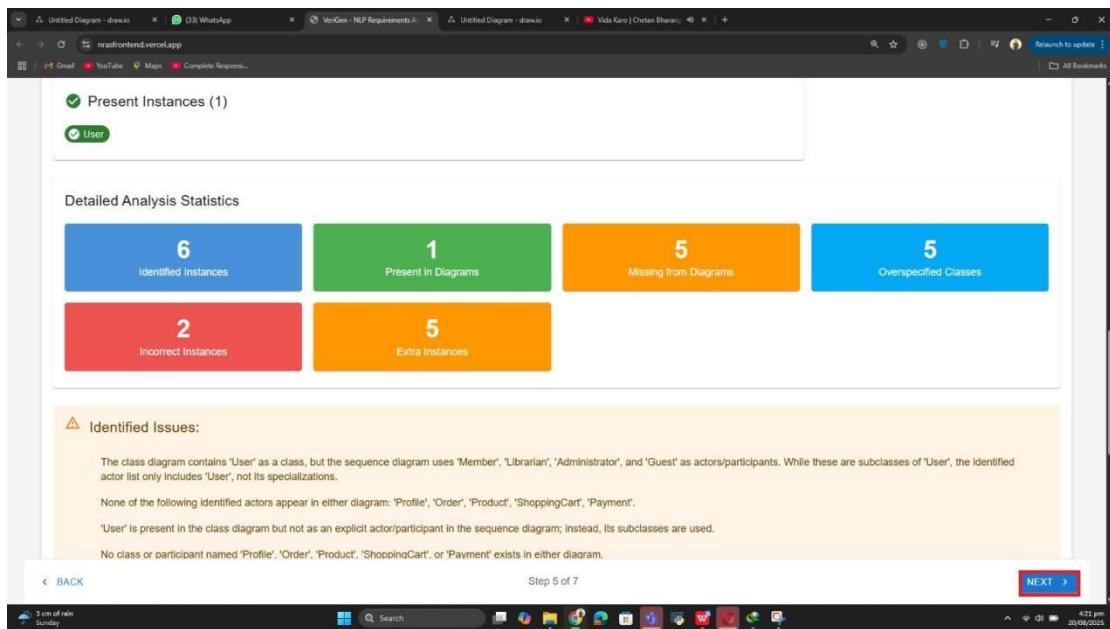
- Shows Verification Results dashboard with scores and statistics



- Shows detailed breakdown of Incorrect, Extra, and Present Instances.



- Shows Detailed Analysis Statistics and Identified Issues.

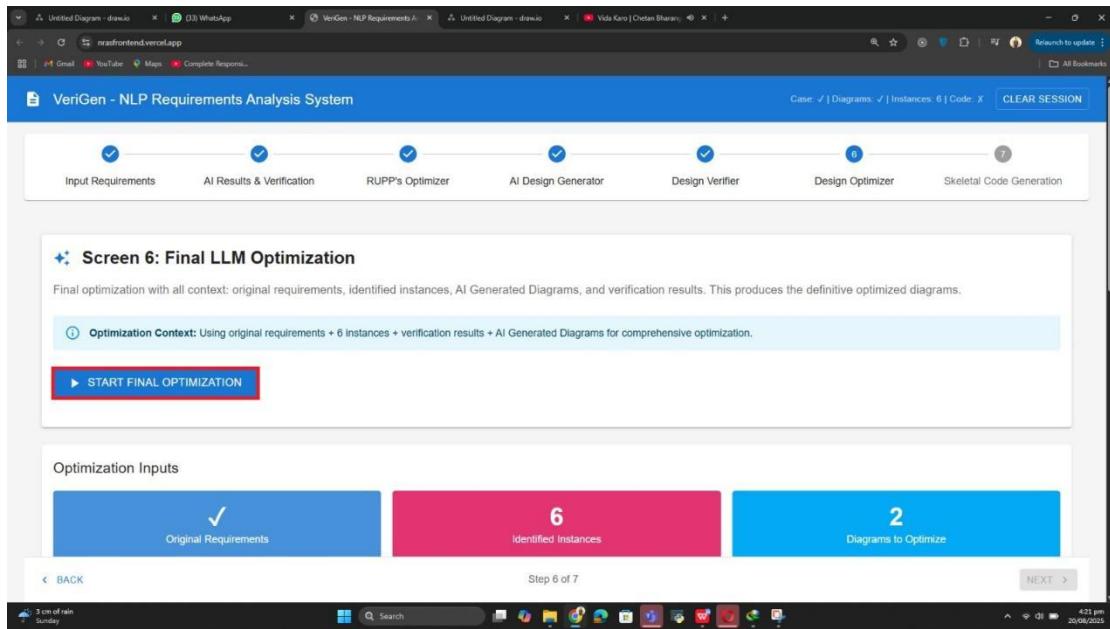


## STEP 6: Final LLM Optimization

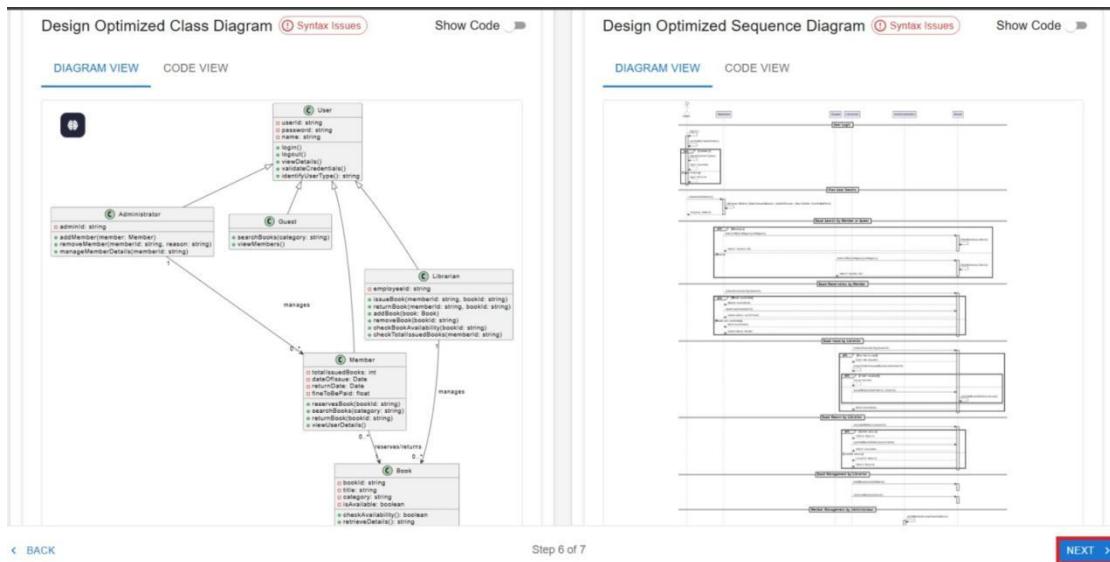
1. System displays "Screen 6: Final LLM Optimization"
2. Click the [► START FINAL OPTIMIZATION] button
3. Review the optimized diagrams:
  - Design Optimized Class Diagram
  - Design Optimized Sequence Diagram
4. Click the [NEXT →] button

**Result:** System generates final optimized diagrams with all corrections applied

- Shows Screen 6 with optimization inputs and Start Final Optimization button.



- Shows final optimized Class Diagram and Sequence Diagram side-by-side.



## STEP 7: Skeletal Code Generation

- System displays "Stage 7: Skeletal Code Generation"
- Configure Project Name and Package Name (optional)
- System automatically starts [ GENERATING JAVA CODE...]
- Review generated Java files:

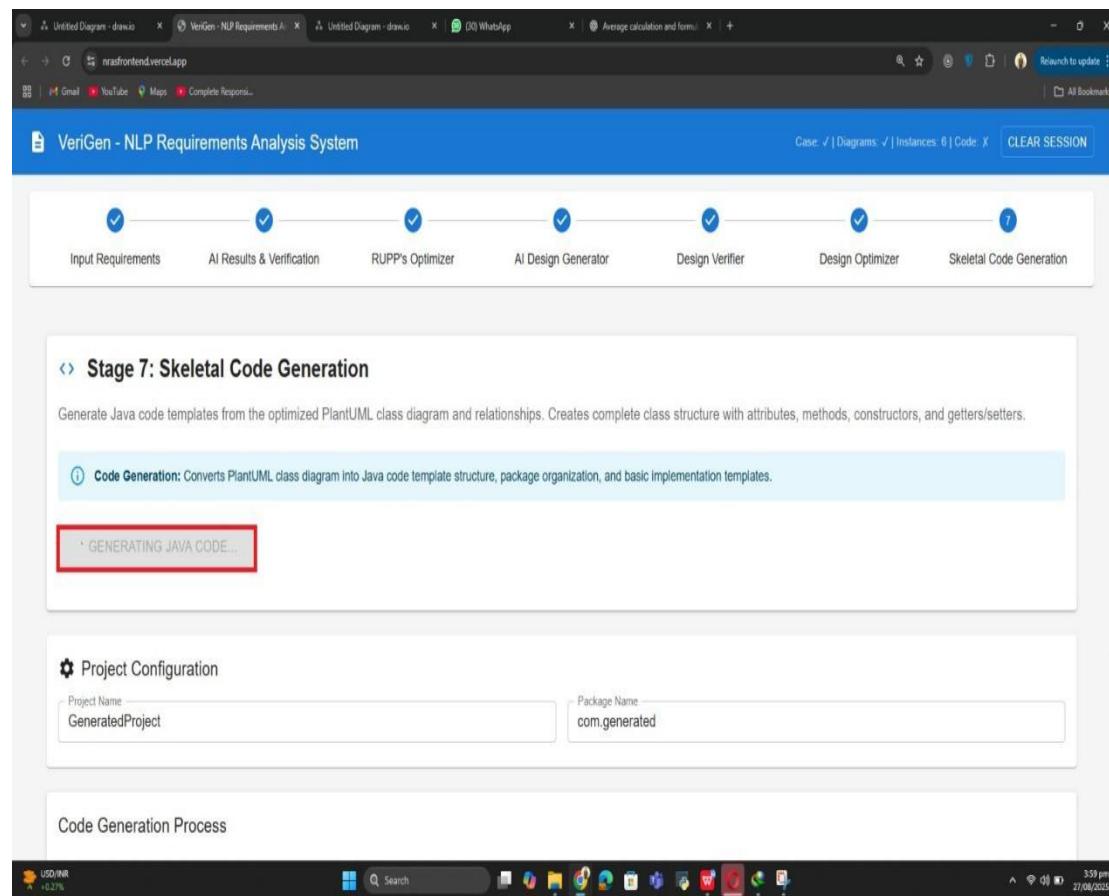
- USER.JAVA
- ADMINISTRATOR.JAVA

- GUEST.JAVA
- LIBRARIAN.JAVA
- MEMBER.JAVA
- BOOK.JAVA

## 5. Click [ DOWNLOAD PROJECT] to download complete project as ZIP file

**Result:** System generates 6 Java class files with skeletal code (package, attributes, constructors, getters/setters)

- Shows Stage 7 with code generation process and project configuration.



- Shows Generated Java Files with code preview and download options.

<> Generated Java Files (6)

[Download Project](#)

USER.JAVA ADMINISTRATOR.JAVA GUEST.JAVA LIBRARIAN.JAVA MEMBER.JAVA BOOK.JAVA



```
1 package com.generated.model;
2
3 public class User
4 {
5
6     // Attributes
7     private String userId;
8     private String password;
9     private String name;
10
11    // Default constructor
12    public User() {
13        // TODO: Initialize default values
14    }
15
16    // Parameterized constructor
17    public User(String userId, String password, String name) {
18        this.userId = userId;
19        this.password = password;
20        this.name = name;
21    }
22
23    // Getters and Setters
```