Application Exercise 1

Functional Requirements:

- User must be able to input mathematical and logical operands and operations
- System must display simplified expressions to user
- User must be able to visualize functions or equalities

Non-Functional Requirements:

- System must be extensible to new expressions and formulas
- System must be scalable to handle growing number of users

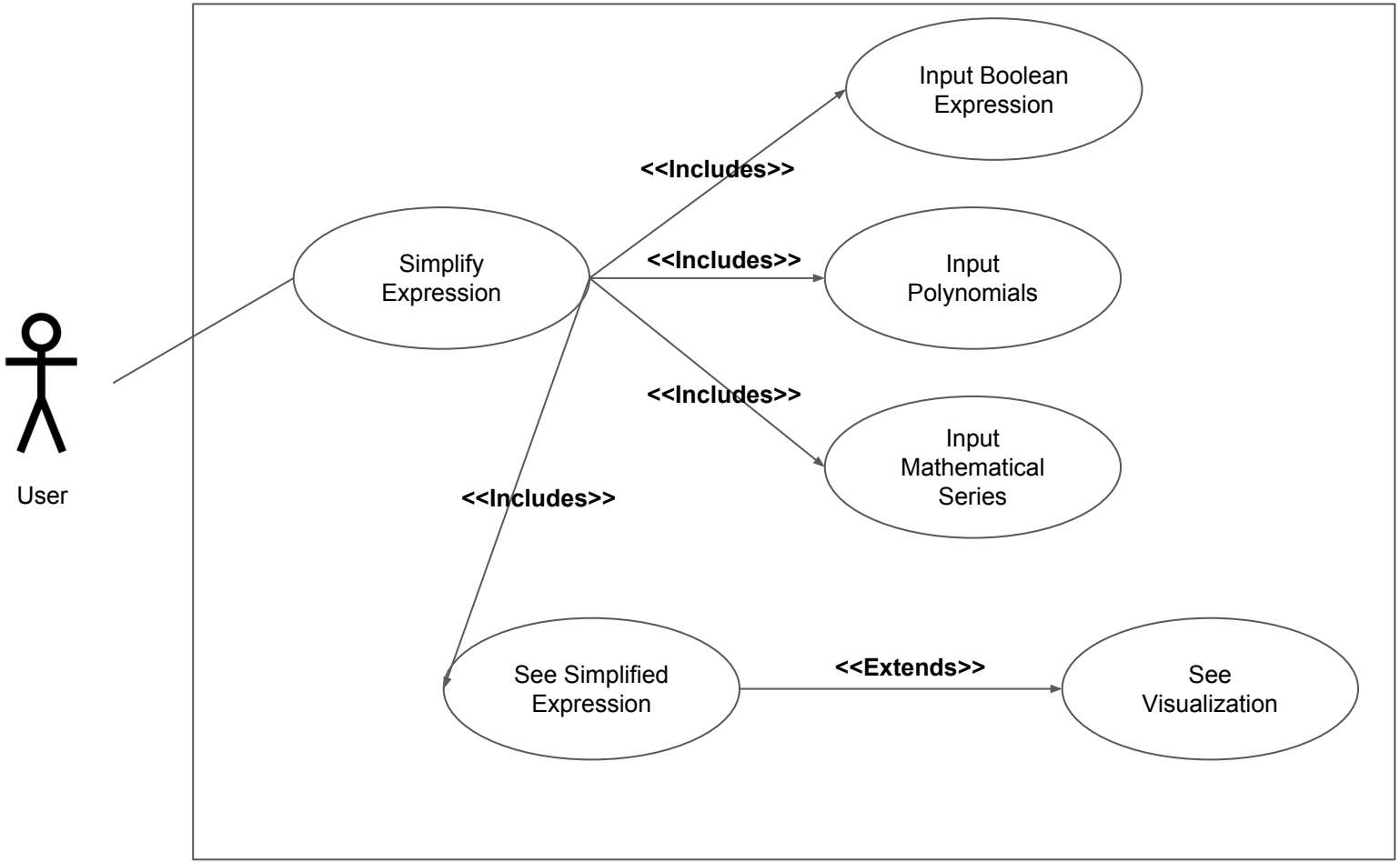
Constraints:

- Must be accessible in web browser

Questions for Client:

Does the tool support...

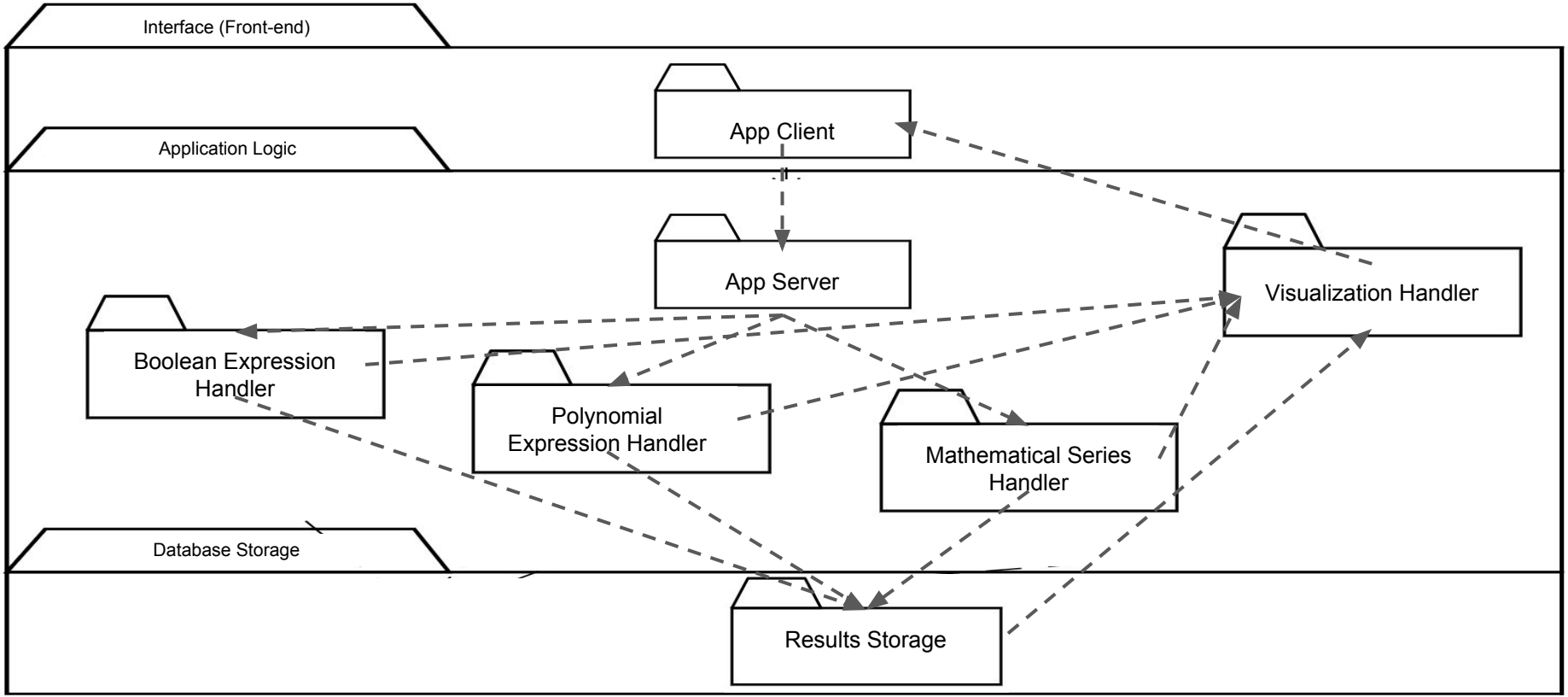
- Integrations/Differentiation?
- Differential equations?
- Probability expressions?
- Systems of equations?
- Complex numbers?
- Trigonometric identities?
- Laplace transforms?
- 3D visualization?


User
Simplify Expression
Input Boolean Expression
Input Polynomials
Input Mathematical Series
See Simplified Expression
See Visualization
<<Includes>>
<<Includes>>
<<Includes>>
<<Includes>>
<<Extends>>

Use Case Descriptions

Use Case Name	Input Boolean Expression	See Visualizations
Participating Actors	Tool user (e.g. educator, student)	Tool user (e.g. educator, student)
Entry Condition	User opens web app	User opens web app
Exit Condition	User clicks to evaluate expression	User receives desired visualizations / error feedback
Flow of Events	1. User enters operands and operators to form desired boolean expression 2. User clicks to evaluate expression 3. Depending on results/feedback the user may repeat 1 and 2	1. User enters operands and operators to form desired formulas 2. User clicks to evaluate expression 3. Depending on returned visualization, the user may repeat 1 and 2
Exceptional Cases	System does not have a specific operand or operator for the user System does not support rules needed for simplification / visualization	System does not support visualization

Application Exercise 2



Subsystem Identification Descriptions

Using a 3-tier architectural style to separate client, application logic and storage.

Subsystems	**Purpose**	**Operations**
Interface Tier: App Client Subsystem	Users inputs and iterates on expression based on outputs and feedback messages.	Get user input, post input to app server, display help/feedback/error to user, get output visualization displays outputs/visualizations
Application Logic Tier: App Server Expression Handlers ● Boolean ● Polynomial ● Series	Get input from client and delegate to respective subsystem. Expression handlers are where domain-specific operand/operator identification takes place. Expression simplification laws are applied here. Results are sent to visualization handler and database storage.	Classify user input, delegate to domain subsystem Interpret expression, apply simplification algorithms, send errors/output to visualization handler and database storage.
Application Logic Tier: Visualization Handler	Gets simplified expression or user error/feedback from expression handlers. Visualizes output.	Get message from expression handlers, apply visualization libraries, send display to App Client
Database Storage Tier: Results Storage	Stores results to return previously computed expressions without transformations.	Get simplified expression from expression handlers, store into database, and send stored outputs to visualizer.