Design patterns
Identify the use of three different design patterns in your project. Describe the context and use of each pattern and draw UML diagrams of how you implemented each pattern.

Design pattern UML diagrams
Here are UML diagrams of the patterns taken from the lecture slides:

- **Strategy**

- **Observer**
**Decorator**

- Component
  - methodA()
  - methodB()
  - // other methods

- ConcreteComponent
  - methodA()
  - methodB()
  - // other methods

- Decorator
  - methodA()
  - methodB()
  - // other methods

- ConcretaDecoratorA
  - Component wrappedObj
  - methodA()
  - methodB()
  - newBehavior
  - // other methods

- ConcretaDecoratorB
  - Component wrappedObj
  - Object newState
  - methodA()
  - methodB()
  - // other methods

**Factory Method**

- Product

- Creator
  - factoryMethod()
  - anOperation()

- ConcretaProduct

- ConcretaCreator
  - factoryMethod()
Iterator

Composite

Clients also add(), remove(), and getChild(). With a Composite class, a Client can call add(), remove(), or operation() for a Leaf node. A Leaf has no children.
Your implementation of three design patterns
For each of your three chosen design patterns, include in a written report:

- Describe the context in which you used the pattern. What functionality is provided by the code that implements the pattern?

- Draw a UML diagram of your implementation of the pattern. Identify how each class corresponds to the generic class of the pattern. In other words, how does your diagram mirror the diagram of the pattern as shown above? For example, if you chose the Observer Design Pattern, which of your classes correspond to the Subject interface, the ConcreteSubject class, the Observer interface, and the ConcreteObserver class?

- Include your C++ source files that contain your implementation of the design pattern.

If you are not sufficiently far along in coding your project, then think about what design patterns would be appropriate for you to use, and then write some prototype C++ code for your project that implements the patterns.

What to turn in
Each team should submit into Canvas a zip file containing your report and source files:
Assignment #6: Design Patterns

Rubric
Your assignment will be graded according to these criteria:

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Max points</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>First design pattern</strong></td>
<td>35</td>
</tr>
<tr>
<td>• Description of the context of the use of the pattern.</td>
<td>15</td>
</tr>
<tr>
<td>• UML diagram of the pattern’s implementation.</td>
<td>15</td>
</tr>
<tr>
<td>• C++ source code containing the implementation.</td>
<td>5</td>
</tr>
<tr>
<td><strong>Second design pattern</strong></td>
<td>35</td>
</tr>
<tr>
<td>• Description of the context of the use of the pattern.</td>
<td>15</td>
</tr>
<tr>
<td>• UML diagram of the pattern’s implementation.</td>
<td>15</td>
</tr>
<tr>
<td>• C++ source code containing the implementation.</td>
<td>5</td>
</tr>
<tr>
<td><strong>Third design pattern</strong></td>
<td>35</td>
</tr>
<tr>
<td>• Description of the context of the use of the pattern.</td>
<td>15</td>
</tr>
<tr>
<td>• UML diagram of the pattern’s implementation.</td>
<td>15</td>
</tr>
<tr>
<td>• C++ source code containing the implementation.</td>
<td>5</td>
</tr>
</tbody>
</table>