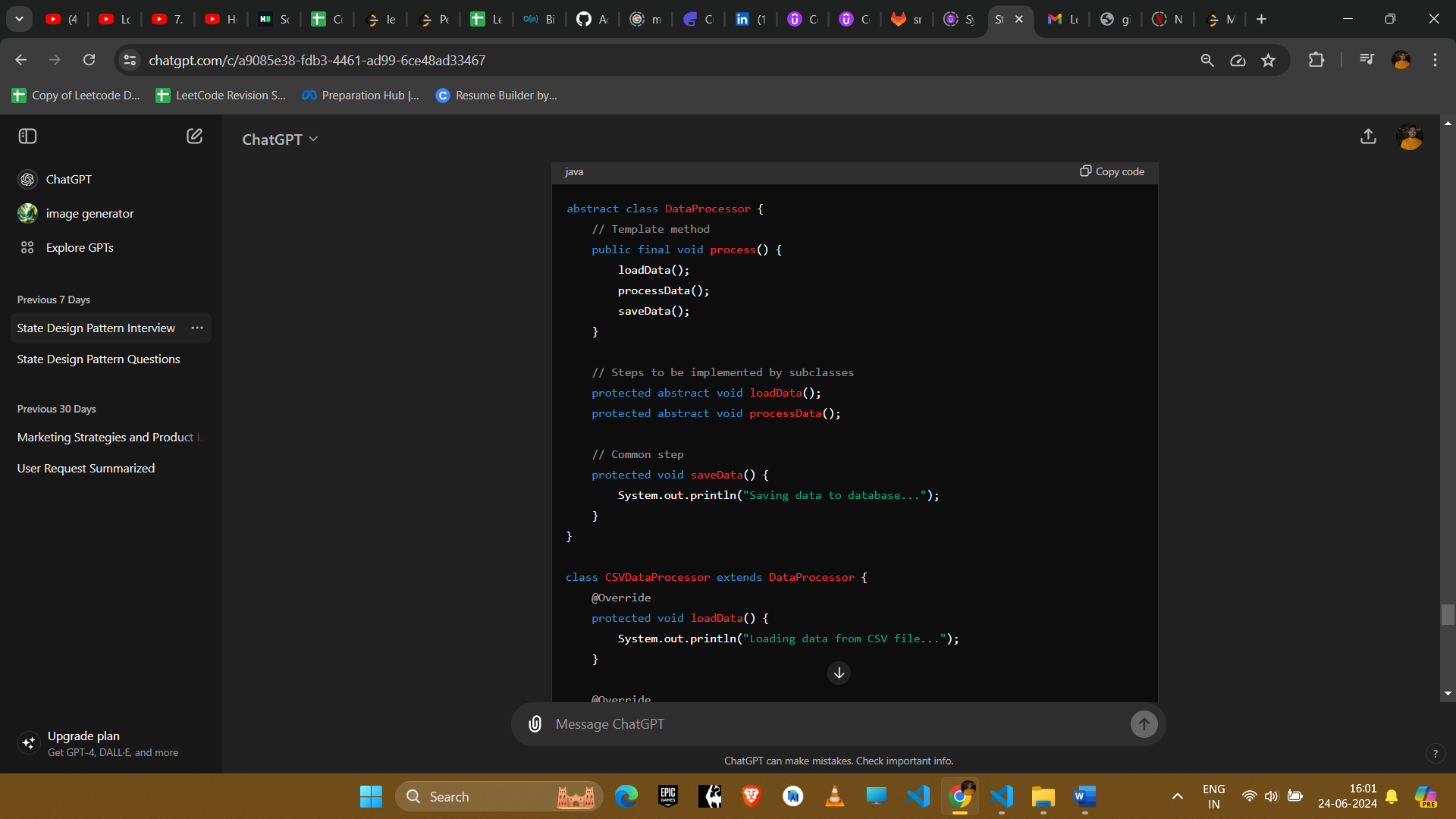
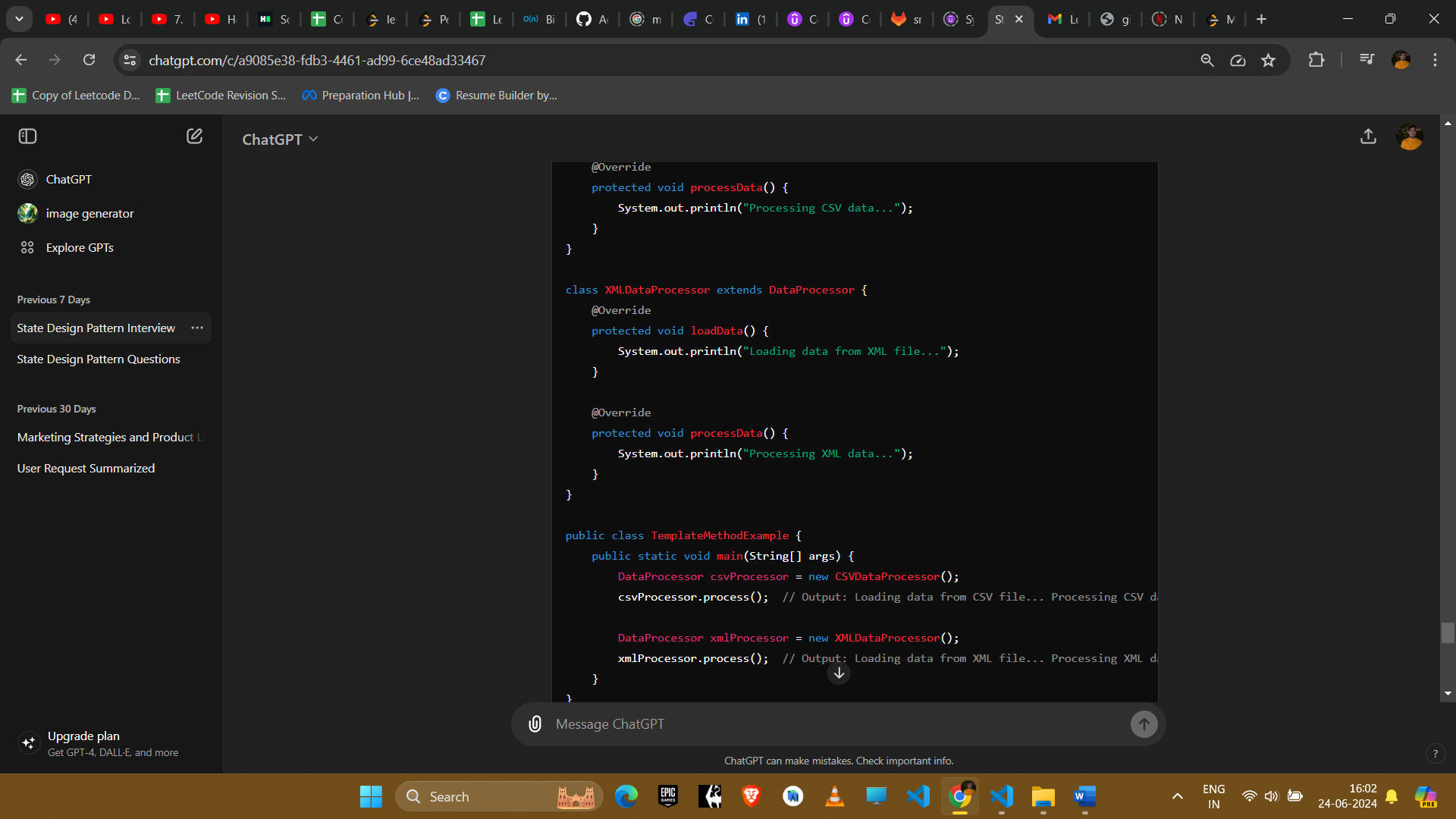
**Template Method Design Pattern**

The Template Method design pattern is a behavioural pattern that defines the skeleton of an algorithm in a base class while allowing subclasses to override specific steps of the algorithm without changing its structure. This promotes code reuse and enforces a consistent structure for certain processes





### Explanation

1. **Abstract Class**: DataProcessor defines the template method process that outlines the algorithm's structure.
2. **Abstract Methods**: loadData and processData are abstract methods that subclasses must implement.
3. **Concrete Classes**: CSVDataProcessor and XMLDataProcessor implement the abstract methods, providing specific behaviors for loading and processing data.

### Example Uses in Amazon Interviews

#### 1. **File Processing System**

* **Scenario**: Processing different types of files (e.g., CSV, XML, JSON) with common steps like loading, processing, and saving.
* **Implementation**: Define a template method in a base class that calls abstract methods for loading and processing data. Subclasses implement these steps for specific file types.

#### 2. **Data Import/Export**

* **Scenario**: Importing and exporting data in different formats (e.g., database, flat file, API).
* **Implementation**: Use the Template Method pattern to define the sequence of steps for import/export operations. Subclasses handle specific formats.

#### 3. **Task Scheduling System**

* **Scenario**: Different tasks (e.g., sending emails, generating reports) need to follow a common scheduling and execution process.
* **Implementation**: Create a base class with a template method defining the scheduling process. Subclasses implement the task-specific execution logic.

#### 4. **Algorithm Frameworks**

* **Scenario**: Implementing various algorithms with a common structure but different details (e.g., sorting algorithms, search algorithms).
* **Implementation**: Use the Template Method pattern to define the high-level algorithm structure. Subclasses implement specific steps.

#### 5. **UI Component Rendering**

* **Scenario**: Rendering different types of UI components (e.g., buttons, text fields, drop-downs) with common steps for setup and teardown.
* **Implementation**: Define a template method for rendering that includes steps like setup, drawing, and teardown. Subclasses implement the drawing step for specific components.

### Conclusion

The Template Method design pattern is useful for defining the outline of an algorithm while allowing subclasses to provide specific implementations for certain steps. It promotes code reuse, ensures a consistent structure, and is particularly useful in scenarios where an algorithm consists of invariant parts with some variable steps.