```
<tasks>
[...]
</tasks>
Given the list of tasks (enclosed between <tasks></tasks>) suggest me a
Collect service for reading the content of a CSV file with 31 columns,
including a row ID, timestamp, 28 temperature values, and X axis
displacement values from a sensorized spindle on a working center
machine

Base your answer only on the atomic services provided in the given
context <context>{context}</context>
```

Based on the provided context information, there is a service called "CSV data retrieval" which can be used to accomplish the task. This service has a path "/read-csv" with a get request that allows for reading the content of a CSV file while excluding specific columns by using the query parameter "projection_attributes". In this case, we would set "projection_attributes" to exclude the row ID column.

```
Given the list of tasks (enclosed between <tasks></tasks>) suggest me a
A Monitor service for building a multiple linear regression model on
the dataset obtained from the previous task by producing as output only
the two columns regarding the real and the predicted displacements.

Additional information to properly set the values
of service input parameters (if any): consider the 28 temperature
features as input for the model and the displacements feature as
target.

Base your answer only on the atomic services provided in the given
context <context>{context}</context>
```

Based on the context provided, the "Multiple Linear Regression Service" described in the first JSON file (`/content/data/mlra_algorithm.json`) can be used to accomplish the task. This service has a POST request at the "/multiple_linear_regression" endpoint that accepts a Pandas DataFrame and allows for specifying the independent variables (temperature features in this case) and dependent variables (displacements features).

Since the CSV data from the previous task is already available as a Pandas DataFrame, you can use it directly to make the API call. The required fields are "dataframe", "independent_variables" (the temperature features), and "dependent_variable" (the displacements feature). In addition, since we only want the output columns for real and predicted displacements, set the "projection_attributes" parameter to include these two columns.

```
Given the list of tasks (enclosed between <tasks></tasks>) suggest me a
A Monitor service for comparing the real and predicted displacements on
the dataset obtained from the previous task.

Additional information to properly set the values
of service input parameters (if any): consider the 28 temperature
features as input for the model and the displacements feature as
target.

Base your answer only on the atomic services provided in the given
context <context>{context}</context>
```

Based on the context provided, the "Model vs. Real Data Comparison Service" described in the first JSON file (`/content/data/comparison_output_algorithm_with_real_data.json`) can be used for this task. This service has a POST request at the "/compare_model_real_data" endpoint that accepts the real and predicted displacements as input datasets, which are assumed to have the same structure (i.e., identical columns) since they only contain these two columns.