About The Project

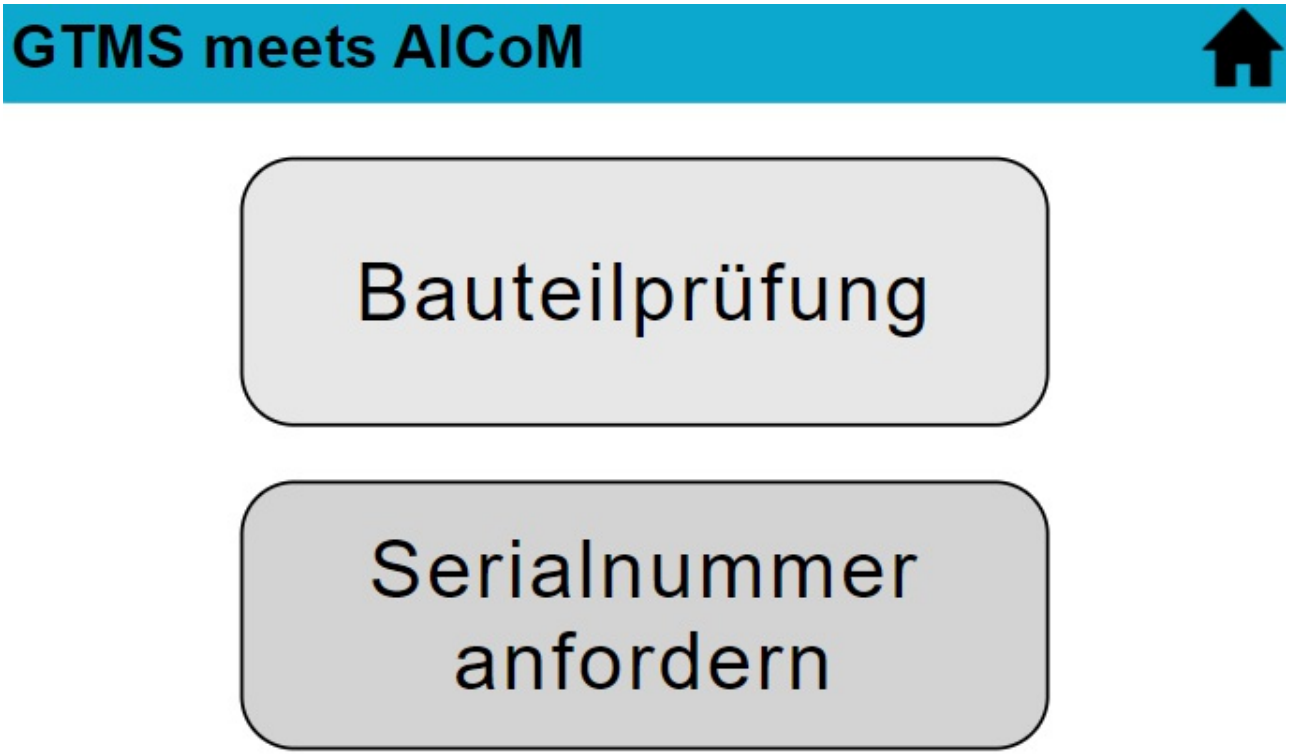
This project is a German web application developed for a research project Artificial Intelligence Controlled Milling (AICoM). The app is used for inspection of components in metalworking process for the manufacturing industry.

Visit this link for more info about the research project: AICoM (https://www.ptw.tu-darmstadt.de/forschung_ptw/tec/aktuelle_projekte_tec/aicom_1/index.en.jsp)

Application Flow 1

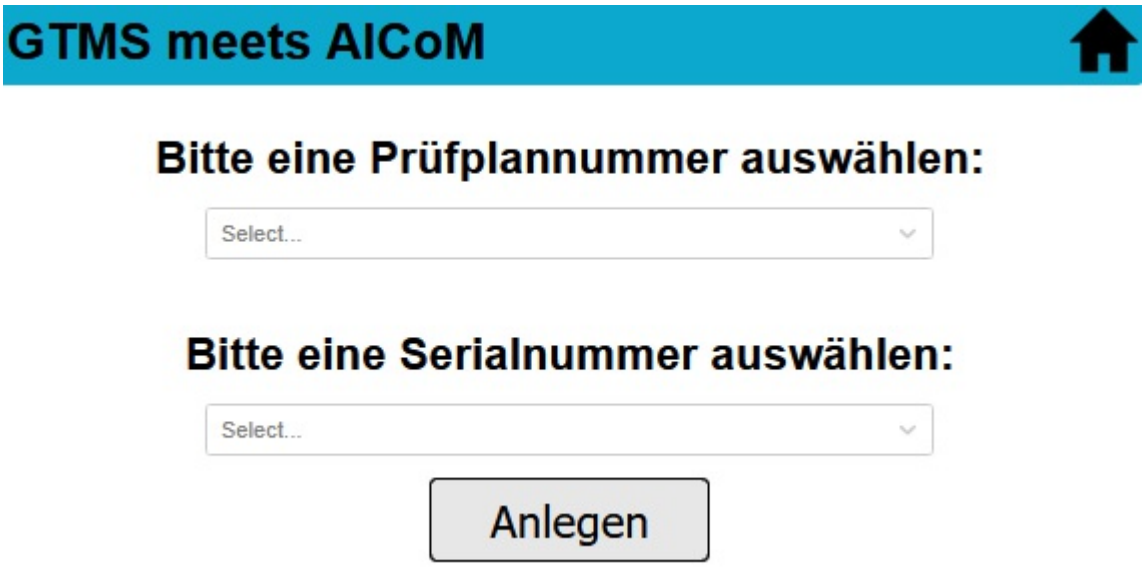
First function: Inspection of a component

Homepage



On every webpage of the app, there are a home button on the top right of the page. By clicking it, it goes to the homepage. On the homepage, there are two buttons "Bauteilprüfung" (inspection of a component) and "Serialnummer anfordern" (create a serial number). The first button "Bauteilprüfung" is used to inspect a component. By clicking it, users are directed to the first page.

First page - Selection of Inspection Number and Serial Number



Users have to choose an inspection number and a serial number of the component, and click "Anlegen" to create a measurement. Now on the second page, the app will show all the measurement features saved in DB.

Second page - Input Actual Value for Measurement Features

GTMS meets AICoM

Zeichnung öffnen

Prüfplan	Pos.	Formelement	Prüfmerkmal	Messmittel	Min.	Max.	Soll	IST-Wert	Bewertung	Bemerkung
1000	1	4_3_1	Nut 2mm		1,89	2	2	2	i.O	
1000	2	4_2_1	Nut 3mm		2,98	3	3	3	i.O	
1000	3	4_1_1	Nut 6mm		5,98	6	6	6	i.O	
1000	4	5_1_1	Tasche 20mm R5		19,92	20	20	20	i.O	
1000	5	5_2_1	Tasche 20mm R3		19,92	20	20	20	i.O	
1000	6	5_3_1	Tasche 10x20		9,91	10	10	8	n.i.O	
1000	7	5_6_1	Tasche mit Steg		11,94	12	12	11	n.i.O	
1000	8	5_4_1	D=15mm(6tief)		14,74	15	15	15	i.O	
1000	9	5_5_1	D=15mm(10tief)		14,8	15	15	15	i.O	
1000	10	5_7_1	D=15mm(mitte D=7.5mm)		14,85	15	15	15	i.O	
1000	11	5_11_1	D=8mm(6tief)		7,95	8	8	8	i.O	
1000	12	2_1_1	D=8mm(8tief)		7,92	8	8	8	i.O	
1000	13	5_10_1	D=8mm(10tief)		7,92	8	8	8	i.O	
1000	14	5_12_1	D=5mm		4,77	5	5	0	n.i.O	
1000	15	2_2_1	Gewinde M6(9tief)		6	6	6	0	n.i.O	
1000	16	2_3_1	Gewinde M6(11tief)		6	6	6	0	n.i.O	

Zurück Speichern

Users can click the button "Zeichnung öffnen" to see the image of the component. In this page, users are required to fill in the "IST-Wert" (actual value) for each feature.
The app evaluates automatically the input actual value by checking whether it is within the "Max." (maximum value) and "Min." (minimum value).
If it is within the range, the column "Bewertung" (evaluation) shows "i.O" (okay). If it is not within the range, "n.i.O" (not okay) is shown

and the row is red. Users can also input remarks in the column "Bemerkung".
Users can then click "Speichern" to save the measurement data. After clicking save, it goes to the final page. Or users can also click the button "Zurück" to go back to the previous page.

Final page - Presentation of Measurement Results

GTMS meets AICoM

Prüfplan	SerialNr.	Position	Formelement	Messmittel	IST-Wert	Bewertung	Bemerkung
1000	0	1	4_3_1	Nut 2mm	2	i.O	
1000	0	2	4_2_1	Nut 3mm	3	i.O	
1000	0	3	4_1_1	Nut 6mm	6	i.O	
1000	0	4	5_1_1	Tasche 20mm R5	20	i.O	
1000	0	5	5_2_1	Tasche 20mm R3	20	i.O	
1000	0	6	5_3_1	Tasche 10x20	8	n.i.O	
1000	0	7	5_6_1	Tasche mit Steg	11	n.i.O	
1000	0	8	5_4_1	D=15mm(6tief)	15	i.O	
1000	0	9	5_5_1	D=15mm(10tief)	15	i.O	
1000	0	10	5_7_1	D=15mm(mitte D=7.5mm)	15	i.O	
1000	0	11	5_11_1	D=8mm(6tief)	8	i.O	
1000	0	12	2_1_1	D=8mm(8tief)	8	i.O	
1000	0	13	5_10_1	D=8mm(10tief)	8	i.O	
1000	0	14	5_12_1	D=5mm	0	n.i.O	
1000	0	15	2_2_1	Gewinde M6(9tief)	0	n.i.O	
1000	0	16	2_3_1	Gewinde M6(11tief)	0	n.i.O	

Schließen

The final page shows the evaluated measurement results. Users can click "Schließen" to close it and users are directed to homepage and can create another measurement.

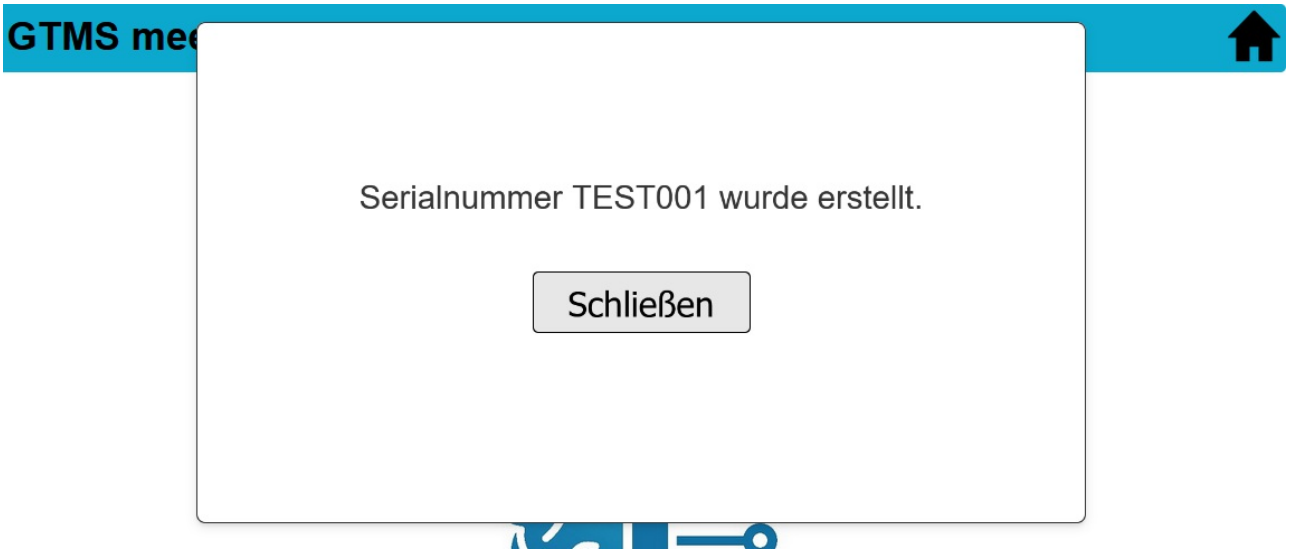
Application Flow 2

Second function: Create a serial number

Homepage

On the homepage, the second button "Serialnummer anfordern" is used to create a new serial number for a component.

After clicking it, users are directed to the next page to choose the components that needs a serial number. After selection, click on "Anlegen" to create a new serial number.



The app shows a message box that a new serial number XXX was created.

Table of Contents

Getting Started

This app is sharing the same database (DB) with Gühring Tool Management Software (https://guehring.com/en/service/digital-services/gtms/) (GTMS). First of all, users have to maintain the data of measurement features and tolerance of the components using GTMS.

To use this app:

1. Refer to the user manual "Bauteilpruefung_GTMS-Anleitung" in the folder D:\GTMS\Batueilpruefung for a step by step guide to save the prerequisite measurement features and their tolerances in GTMS.
2. Users can see the app on the following URL: http://localhost:7000/menu

Technologies Used

- React (https://reactjs.org/docs/getting-started.html)
- JavaScript
- Styled-components for styling (https://styled-components.com/docs)

Environment Setup

- Node.js (version 18.15.0 or higher)
    - Download and run the installer from NodeJS WebSite (https://nodejs.org/en).

ESLint and Prettier Setup

ESLint (https://eslint.org) is a static code analysis tool for identifying problematic patterns found in JavaScript code.

- Install ESLint locally according to ESLint Website (https://eslint.org/docs/latest/use/getting-started):
  `npm init @eslint/config`
- Then, configure ESLint via .eslintrc.js file.
- Refer to the Configure ESLint documentation (https://eslint.org/docs/latest/use/configure/) to learn how to add rules,

environments, custom configurations, plugins, and more.

Prettier (https://prettier.io) is an opinionated code formatter.

- Install Prettier locally accroding to Prettier Website (https://prettier.io/docs/en/install.html):
  `npm install --save-dev --save-exact prettier`
- Then, configure Prettier via .prettierrc.json file to let editors and other tools know you are using Prettier.
- Refer to the Configuration File documentation (https://prettier.io/docs/en/configuration.html) to learn more.

Project Structure

```
├── node_modules/ # Dependencies
├── public/
│ ├── pictures/ # All the pictures used in the app such as logo, home button and components for inspection
│ ├── index.html
├── src/
│ ├── components/
│ │ ├── Header/
│ │ │ ├── Header.js # Header with title of the app and home button
│ │ │ ├── index.js
│ │ ├── Image/
│ │ │ ├── ImageModal.js # Image enlargement upon image click. It is used in the second page of the app (Results.js)
│ │ │ ├── index.js
│ │ ├── Table/
│ │ │ ├── SelectRow.js # Selection of "i.O" (okay) or "n.i.O" (not okay) for the column "Bewertung" (evaluation) in each row of the table in second page. If the measurement feature of a row has no value input, then these selections are active and shown in the row for users to select.
│ │ │ ├── TableAuftragPruefdaten.js # This is the table shown in final page. It shows all the evaluated measurement data of the inspection of a component done by users.
│ │ │ ├── TableAuftragPruefpositionen.js # This is the table shown in second page. It shows all the measurement features of the inspected component. Users have to fill in the "IST-Wert" (actual value) in this table.
│ │ ├── UI/ # Styling of common UI components using styled-components
│ │ │ ├── Button.js # Styles of buttons: small, medium and big
│ │ │ ├── NumberInput.js # Number input for "IST-Wert" (actual value) column in the table of the second page
│ │ │ ├── SelectMenu.js # Select menu in the first page to choose inspection number and serial number or to choose component when creating serial number
│ │ │ ├── Table.js # Table styling for second page and final page
│ │ │ ├── TextInput.js # Text input for "Bemerkung" (remark) colum in the table of the second page
│ │ ├── layouts/
│ │ │ ├── Layout.js # Standard layout for every page
│ │ ├── pages/
│ │ │ ├── Bauteilpruefung/
│ │ │ │ ├── Bauteilpruefung.js # First page for selecting an inspection number and a serial number for inspection of a component
│ │ │ │ ├── index.js
│ │ │ ├── Finalpage/
│ │ │ │ ├── Finalpage.js # Final page to show the saved evaluated measurement results
│ │ │ │ ├── index.js
│ │ │ ├── Menu/
│ │ │ │ ├── Menu.js # Homepage with two buttons with the functions: 1. Inspect a component, 2. Create a serial number
│ │ │ │ ├── index.js
│ │ │ ├── Results/
│ │ │ │ ├── Results.js # Second page for input measurements and every row shows the evaluated results immediately
│ │ │ │ ├── index.js
│ │ │ ├── Serialnummer/
│ │ │ │ ├── Serialnummer.js # The page after clicking the second button on homepage for creating a new serial number
│ │ │ │ ├── index.js
│ ├── .env.local # Configuration option for Backend API PORT
│ ├── App.js # App main component
│ ├── index.css # Standard CSS styling for app
│ ├── index.js # Entry point of app
├── .eslintrc.js # Static code analysis tool
├── .gitignore # Ignore files or folders in Git
├── .prettierrc.json # Code formatter
├── README.md # Documentation
├── jsconfig.json # JavaScript language service
├── package-lock.json # Generated by npm when installing packages
├── package.json # npm dependencies and run scripts
```

User Interface

This app consists of the following main UI components:

- Header: Header with home button on every page.
- Image: Image enlargement upon component click for the button "Zeichnung öffnen" in the second page: A user interface feature or interaction where an image associated with a specific component enlarges or magnifies when the user clicks on (or taps, in touch interfaces) that component.
- Table: Table for users to input measurement and remarks in the second page and show evaluated measurement results in the final page. It contains the measurement details such as "Prüfplan" (inspection number), "Serialnr." (serial number), "Position" (position), "Formelement" (process number), "Messmittel" (measuring tools), "Min." (minimum value), "Max." (maximum value), "Soll" (desired value), "IST-Wert" (actual value), "Bewertung" (evaluation) and "Bemerkung" (remark).
- UI: Styling for common UI components such as Button, Select Input, Number Input, Text Input and Table

Data Management

Data is fetched from a backend API using Fetch API and displayed in the UI components.

1. Fetch inspection plans
    - URL: http://localhost:8003/AuftragPruefplan
    - Method: GET
    - Success Response:
        - Code: 200 OK
        - Content:
            - json

```
[
  {
  "ID": 1,
  "Pruefplannummer": 1000,
  "Pruefplan": "AICoM Original",
  "AnzahlPruefteile": 0,
  "AnzahlPruefteileProzentual": 0,
  "Teilung": 0,
  "ErstUndLetztPruefung": false,
  "Dokument1": "C:\\pictures\\AICoM_Referenzbauteil1.JPG",
  "Dokument2": "",
  "Zusatztext1": "",
  "Zusatztext2": "",
  "Zusatztext3": "",
  "Zusatztext4": "",
  "Zusatztext5": "",
  "AngelegtAm": null,
  "AngelegtVon": ""
  },
  ......
]
```

    - Error Response: - Code: 500 Internal Server Error
        - Content:
            - json

```
{
  "error": "Error Message"
}
```

2. Fetch measurement features
    - URL: http://localhost:8003/AuftragPruefpositionen/:selectedPruefplannummer
    - Method: GET
    - URL Parameters:
        - selectedPruefplannummer (required): The inspection number
    - Success Response:
        - Code: 200 OK

        - Content:
            - json

```
[
  {
    "ID": 17,
    "Pruefplannummer": 1001,
    "Position": 1,
    "Positionstext": "1_1_1",
    "Artikel": "6793",
    "Bezeichnung": "45H7 (+0.025)",
    "MinWert": "45,7",
    "MaxWert": "45,725",
    "Sollwert": "45,7",
    "Endkontrolle": false,
    "KeineWerteingabe": false,
    "Zusatztext1": "Innenmessschraube",
    "Zusatztext2": "",
    "Zusatztext3": "",
    "Zusatztext4": "",
    "Zusatztext5": ""
  },
  ......
]
```

    - Error Response: - Code: 500 Internal Server Error
        - Content:
            - json

```
{
  "error": "Error Message"
}
```

3. Fetch image of a component
    - URL: http://localhost:8003/AuftragPruefplan/:selectedPruefplannummer
    - Method: GET
    - URL Parameters:
        - selectedPruefplannummer (required): The inspection number
    - Success Response:
        - Code: 200 OK
        - Content:
            - json

```
{
  "success": "Copy image file from C:\\pictures\\AICoM_Referenzbauteil1.JPG to D:\\GTMS\\aicom-mw-api\\bauteilpruefung-frontend\\public\\pictures\\1000.jpg successfully."
}
```

    - Error Responses:
        - Code: 404 Not Found
            - Scenario: When trying to access an image which is not available.
                - Content:
                    - json

```
{
  "error": "Image path doesn't exists."
}
```

            - Scenario: When trying to access an image which is not available.

- Content: When trying to access a file that isn't present.
  - json

```
{
  "error": "No such file or directory."
}
```

- Code: 500 Internal Server Error
  - Content:
    - json

```
{
  "error": "Error Message"
}
```

4. Fetch logo
   - URL: http://localhost:8003/readFile/config/logo
   - Method: GET
   - Success Response:
     - Code: 200 OK
     - Content:
       - json

```
{
  "success": "Copy logo file from C:\\pictures\\Aicom_logo.jpg to
D:\\GTMS\\aicom-mw-api\\bauteilpruefung-
frontend\\public\\pictures\\copiedLogo.jpg successfully."
}
```

   - Error Responses:
     - Code: 404 Not Found
       - Scenario: When trying to access an image which is not available.
         - Content:
           - json

```
{
  "error": "Image path doesn't exists."
}
```

       - Scenario: When trying to access an image which is not available.
         - Content: When trying to access a file that isn't present.
           - json

```
{
  "error": "No such file or directory."
}
```

     - Code: 500 Internal Server Error
       - Content:
         - json

```
{
  "error": "Error Message"
}
```

5. Fetch website RGB color codes
   - URL: http://localhost:8003/readFile/config/color
   - Method: GET

- Success Response:
  - Code: 200 OK
  - Content:
    - json

```
{
  "HeaderRot": "12",
  "HeaderGruen": "168",
  "HeaderBlau": "206"
}
```

- Error Response:
  - Code: 500 Internal Server Error
    - Content:
      - json

```
{
  "error": "Error Message"
}
```

6. Fetch components
   - URL: http://localhost:8003/Bauteile
   - Method: GET
   - Success Response:
     - Code: 200 OK
     - Content:
       - json

```
[
  {
  "ID": 1,
  "Bauteil": "AICoM Original",
  "Bauteilvariante": "",
  "Bauteilnummer": "",
  "Bauteiltyp": "",
  "Artikel": "5768",
  "Zeichnungsnummer": "",
  "Zeichnung": "D:\\GTMS\\Bilder\\AICoM_Referenzbauteil1.JPG",
  "Werkstoff": "",
  "Preis": 0,
  "WechselJahrMillionSoll": 0,
  "BruecheJahrMillionSoll": 0,
  "Bauteilplanung": false,
  "BauteilGesamtstueckzahl": 0,
  "BauteilGesamtkosten": 0,
  "KostenProBauteilSoll": 0,
  "Beschaffung": "",
  "Verantwortlich": "Admin",
  "AufbereitungIntern": false,
  "EinstellungIntern": false,
  "Zusatztext1": "Material: AlMg4,5Mn",
  "Zusatztext2": "Geometrie: 200x200x30mm",
  "Zusatztext3": "",
  "Zusatztext4": "",
  "Zusatztext5": "",
  "AngelegtAm": "2022-03-15T11:39:25.540Z",
  "AngelegtVon": "Admin"
  },
  ......
]
```

   - Error Response: - Code: 500 Internal Server Error

        - Content:
            - json

```
{
  "error": "Error Message"
}
```

7. Fetch serial numbers
    - URL: http://localhost:8003/Serialnummern/:pruefplannummer
    - Method: GET
    - URL Parameters:
        - pruefplannummer (required): The inspection number
    - Success Response:
        - Code: 200 OK
        - Content:
            - json

```
[
  {
    "ID": 8,
    "Artikel": "6793",
    "Serialnummer": "V130_2",
    "RFIDCode": "",
    "Anschaffungsdatum": "2023-08-14T08:27:18.407Z",
    "Ausgangswert": 0,
    "Ausgangswert2": 0,
    "Restwert": 0,
    "Restwert2": 0,
    "AnzahlZyklen": 0,
    "Restzyklen": 0,
    "Artikelzustand": "",
    "Lieferantnummer": "",
    "Charge": "",
    "LagerplatzID": 0,
    "Bestand": 0,
    "Stueckliste": "",
    "Stuecklistenvariante": "",
    "Maschine": "",
    "Verschrottet": false,
    "Verschrottungsdatum": null,
    "Bemerkung": "",
    "BuchungsID": 0,
    "Krit1": "",
    "Krit2": "",
    "Krit3": "",
    "Krit4": "",
    "Krit5": "",
    "Krit6": "",
    "Krit7": "",
    "Krit8": "",
    "Krit9": "",
    "Krit10": "",
    "Krit11": "",
    "Krit12": "",
    "Krit13": "",
    "Krit14": "",
    "Krit15": "",
    "Krit16": "",
    "Krit17": "",
    "Krit18": "",
    "Krit19": "",
    "Krit20": "",
    "Einsatzzeit": 0,
    "Testwerkzeug": false
  },
  ......
]
```


- Error Response: - Code: 500 Internal Server Error
  - Content:
    - json


```
{
  "error": "Error Message"
}
```


8. Create measurement results
   - URL: http://localhost:8003/AuftragPruefdaten/createNewRecords
   - Method: POST
   - Body Parameters:
     - auftragPruefpositionen (required): Array of measurement features
     - bauteilnummer (required): Serial number

- Success Response:
  - Code: 200 OK
    - Content:
      - json

```
[
  {
    "ID": 1,
    "Pruefplannummer": 1000,
    "Position": 1,
    "Positionstext": "4_3_1",
    "Artikel": "5768",
    "Bezeichnung": "Nut 2mm",
    "MinWert": "1,89",
    "MaxWert": "2",
    "Sollwert": "2",
    "Endkontrolle": false,
    "KeineWerteingabe": false,
    "Zusatztext1": "",
    "Zusatztext2": "",
    "Zusatztext3": "",
    "Zusatztext4": "",
    "Zusatztext5": "",
    "value": "2",
    "bemerkung": "",
    "bewertung": "i.O"
  },
  ......
]
```

- Error Response: - Code: 500 Internal Server Error
  - Content:
    - json

```
{
  "error": "Error Message"
}
```

9. Create a serial number
   - URL: http://localhost:8003/Serialnummern/createNewRecord
     - Method: POST
       - Body Parameters:
         - selectedBauteil (required): The name of component
       - Success Response:
         - Code: 200 OK
           - Content:
             - json

```
{
  "ID": 21,
  "Artikel": "5768",
  "Serialnummer": "TEST001",
  "RFIDCode": "",
  "Anschaffungsdatum": "2023-09-13T13:20:41.470Z",
  "Ausgangswert": 0,
  "Ausgangswert2": 0,
  "Restwert": 0,
  "Restwert2": 0,
  "AnzahlZyklen": 0,
  "Restzyklen": 0,
  "Artikelzustand": "",
  "Lieferantnummer": "",
  "Charge": "",
  "LagerplatzID": 0,
  "Bestand": 0,
  "Stueckliste": "",
  "Stuecklistenvariante": "",
  "Maschine": "",
  "Verschrottet": false,
  "Verschrottungsdatum": null,
  "Bemerkung": "",
  "BuchungsID": 0,
  "Krit1": "",
  "Krit2": "",
  "Krit3": "",
  "Krit4": "",
  "Krit5": "",
  "Krit6": "",
  "Krit7": "",
  "Krit8": "",
  "Krit9": "",
  "Krit10": "",
  "Krit11": "",
  "Krit12": "",
  "Krit13": "",
  "Krit14": "",
  "Krit15": "",
  "Krit16": "",
  "Krit17": "",
  "Krit18": "",
  "Krit19": "",
  "Krit20": "",
  "Einsatzzeit": 0,
  "Testwerkzeug": false
}
```


- Error Response: - Code: 500 Internal Server Error
  - Content:
    - json


```
{
  "error": "Error Message"
}
```


Styling

Styling is done using styled-components (https://styled-components.com/docs/basics#installation). It utilises tagged template literals to style the components.

Below is an example from styled-components docs. It creates two simple components, a wrapper and a title, with some styles attached to it:

```
// Create a Title component that'll render an <h1> tag with some styles
const Title = styled.h1`
  font-size: 1.5em;
  text-align: center;
  color: #bf4f74;
`;

// Create a Wrapper component that'll render a <section> tag with some styles
const Wrapper = styled.section`
  padding: 4em;
  background: papayawhip;
`;

// Use Title and Wrapper like any other React component – except they're styled!
render(
  <Wrapper>
    <Title>Hello World!</Title>
  </Wrapper>
);
```

Troubleshooting

If you encounter any issues, try the following:

- Restart the backend API "AICoM Bauteilpruefung Backend API" at Windows Task Scheduler.

For more support, please contact software developer Xiang Yi Kiong (xiangyi.kiong@guehring.de).

Resources

- React Documentation (https://reactjs.org/docs/getting-started.html)
- Styled-components Documentation (https://styled-components.com/docs)

Contact

Xiang Yi Kiong (xiangyi.kiong@guehring.de)