

# SWO5 - UE02 Dokumentation

Aufwand: 14h

## Contents

1. Architekturbeschreibung:.....	2
1.1 Domäne .....	3
1.2. Logic .....	4
1.3 Kleine Änderungen.....	4
2. Quellcode .....	4
3. Unit-Tests .....	5
3. Inbetriebnahme .....	6
4. Benutzerdokumentation .....	7
4.1. Übersicht .....	7
4.2. Filter.....	8
4.3. Diagramm-Ansicht .....	10
4.4. Tabellen-Ansicht .....	11
4.5. Export-Funktion .....	12
Abbildung 1: Klassendiagramm.....	2
Abbildung 2: ER-Diagramm.....	3
Abbildung 3: Unit Tests .....	5
Abbildung 4: How-To Datenbank hinzufügen .....	6
Abbildung 5: How-to: Daten zur Datenbank hinzufügen.....	6
Abbildung 6: UI - Übersichts-Tab .....	7
Abbildung 7: UI - Bundesland Dropdown .....	8
Abbildung 8: UI - Bezirk Dropdown .....	8
Abbildung 9: UI - Zeitraum Dropdown .....	9
Abbildung 10: UI Datumsauswahl .....	9
Abbildung 11: UI - Diagrammansicht - Gesamt.....	10
Abbildung 12: UI - Diagrammansicht - Infiziert .....	11
Abbildung 13: UI – Tabellenansicht.....	11
Abbildung 14: Export Dialog .....	12
Abbildung 15: Export File .....	12
Abbildung 16: Export Ergebnis.....	13

## 1. Architekturbeschreibung:

Das Projekt orientiert sich stark an demjenigen Projekt, welches im Laufe der Übung durchbesprochen wurde, weshalb der Großteil der Struktur gleich ist. Das entworfene Klassendiagramm sieht wie folgt aus:

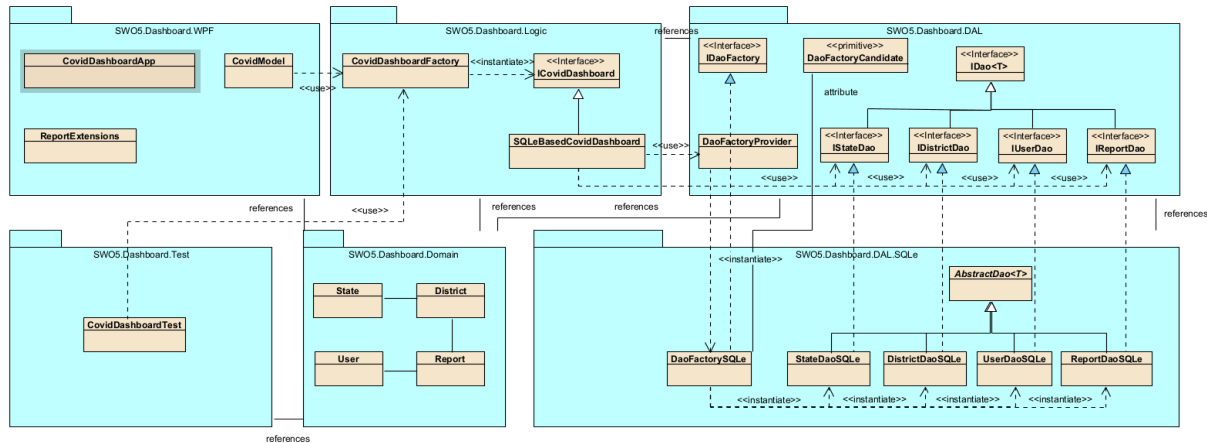


Abbildung 1: Klassendiagramm

Die zwei größten Änderungen stellen die Komponenten „Domain“ und „Logic“ dar, was auch Sinn macht, da das grundlegende Gerüst gleich ist, während sich aber die Daten und die Anwendungsfälle ändern.

Zudem habe ich mich dazu entschieden die Verbindung des DAL-Package mit der Implementierung wie in der Übung zu gestalten, weil ich das Konzept von Attributes und Assemblies verinnerlichen wollte. Ich kann mir vorstellen, dass derartige Flexibilität, auch wenn in diesem Fall nicht gebraucht, sehr hilfreich sein kann.

Dadurch, dass sich Domäne und Logik am meisten verändert haben, möchte ich in den nächsten Kapiteln noch näher auf meine Entscheidungen eingehen.

### 1.1 Domäne

Um die Entscheidungen besser erklären zu können, möchte ich kurz das Datenbankschema illustrieren, da die Klassen in C# ähnlich aufgebaut sind:

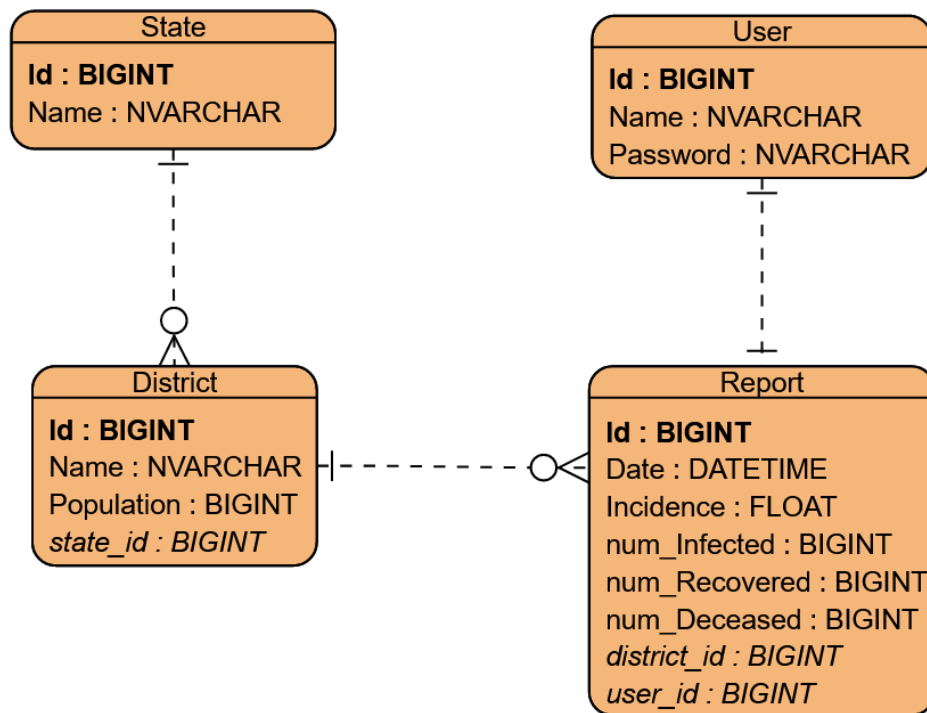


Abbildung 2: ER-Diagramm

Auf den ersten Blick wurden alle relevanten Daten in den Tabellen/Klassen gekapselt.

Anfangs dachte ich daran für District einen Composite-Key zu verwenden, was sich aber mit der Lösung aus dem Unterricht nicht gut vereinbaren lässt, da es wünschenswert ist, dass beim Einfügen bzw. Updaten die ID des Elements retourniert wird um dieses lokal zu vervollständigen bzw. eine Rückmeldung zu erhalten, ob die Datenbankbefehle erfolgreich waren oder nicht. Deswegen habe ich schlussendlich die gesamte GKZ (die indirekt das Bundesland enthält) als Key definiert.

Im Zuge meines Designs habe ich mich außerdem dazu entschieden nicht nur die IDs der Fremd-Entitäten in der C# Implementierung zu speichern, sondern vollständige Referenzen auf die entsprechenden Instanzen, um zu jeder Zeit vollständige Informationen zu haben.

## 1.2. Logic

Wie im Unterricht wurde eine entsprechende Factory und ein entsprechendes Interface für alle gekapselten Funktionalitäten erstellt.

Für die User-Verwaltung wurden die Methoden UserExists und HashPassword angefügt, da ich die Passwörter sicher speichern will und ich mir vorstellen kann, dass es relevant ist, ob ein User (mit bestimmtem Namen) bereits existiert.

Außerdem wurde eine Funktion vorgesehen, welche ein gegebenes CSV der AGES ausliest und dessen Daten in die Datenbank schreibt.

Im Vergleich zur letzten Iteration (Übung 1) wurden die Funktionen, welche ein gewisses Zeitfenster betreffen, entfernt, da diese Funktionalität auch einfach in den entsprechenden Funktionen im GUI verwirklicht werden kann, wo ohnehin LINQ benötigt wird.

Hinzugekommen sind jedoch die Unit-Tests für alle Datenbank-Operationen, die Bundesländer und Bezirke betreffen. Bei der letzten Iteration wurden diese bewusst weggelassen, da diese Datensätze normalerweise konstant bleiben. Diese wurden vollständigheitshalber ergänzt, da es bei anderen Personen deswegen zu einem Punkteabzug gekommen ist.

## 1.3 Kleine Änderungen

Ich habe mich, konträr zur Übung, dazu entschieden, dass ich die JOINS wirklich in den Commands/Selects selbst verwirklichen will und nicht nachträglich im Code die Einträge zusammenstöpseln muss. Daraus ergibt sich, dass sämtliche Read-Methoden für District und Report auch in der spezialisierten Klasse genauer ausgeführt werden müssen, da sich die select-statements nicht mehr in AbstractDao zusammenfassen lassen.

Als Lösung dafür würde ich eine dynamische Bindung implementieren, indem ich die generell formulierten Methoden in AbstractDao als virtual kennzeichne und dann in den relevanten Subklassen überschreibe.

## 2. Quellcode

Der Quellcode wurde aufgrund der schierigen Menge an Dateien nicht direkt ins PDF kopiert, weshalb ich hier auf das Visual Studio Projekt verweise.

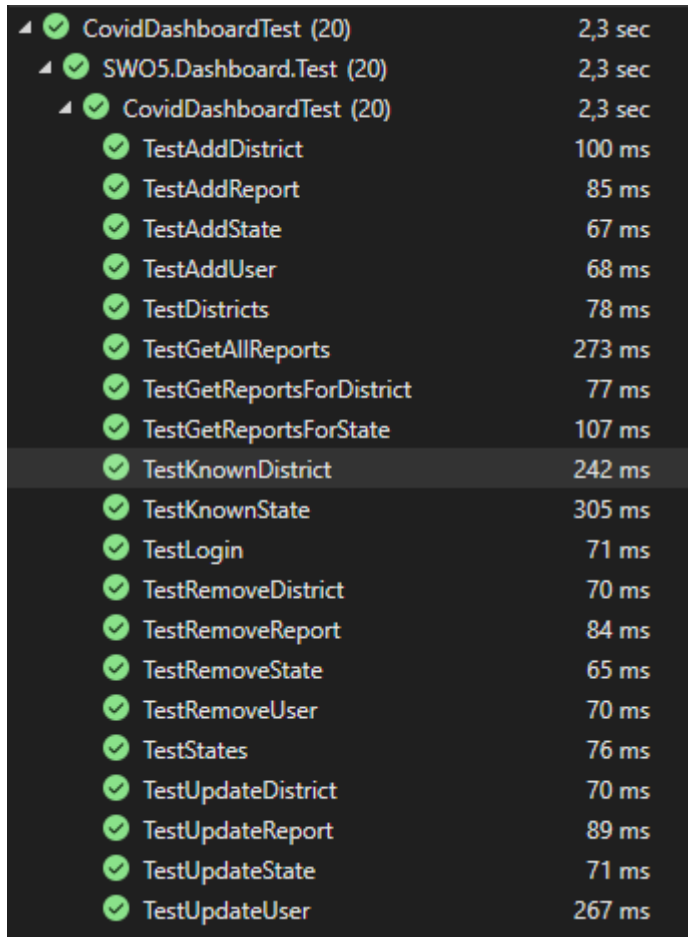
Um den Code in Betrieb zu nehmen, muss vorher noch die Datenbank mithilfe des beigelegten SQL-Scripts erstellt und befüllt werden. Mehr dazu auch in der Benutzerdokumentation.

### 3. Unit-Tests

Ähnlich zum Quellcode werden auch diese im Detail im Code durchgegangen.

Die Testfälle, welche alle Funktionen der Business-Logik abdecken, waren leider teils (nicht vermeidbar) abhängig voneinander, es wurde aber so gut wie möglich versucht, dies zu reduzieren.

Anbei noch ein Screenshot des erfolgreichen Durchlaufs:



CovidDashboardTest (20)	2,3 sec
SWO5.Dashboard.Test (20)	2,3 sec
CovidDashboardTest (20)	2,3 sec
TestAddDistrict	100 ms
TestAddReport	85 ms
TestAddState	67 ms
TestAddUser	68 ms
TestDistricts	78 ms
TestGetAllReports	273 ms
TestGetReportsForDistrict	77 ms
TestGetReportsForState	107 ms
TestKnownDistrict	242 ms
TestKnownState	305 ms
TestLogin	71 ms
TestRemoveDistrict	70 ms
TestRemoveReport	84 ms
TestRemoveState	65 ms
TestRemoveUser	70 ms
TestStates	76 ms
TestUpdateDistrict	70 ms
TestUpdateReport	89 ms
TestUpdateState	71 ms
TestUpdateUser	267 ms

Abbildung 3: Unit Tests

Wie oben bereits erwähnt, wurden die Funktionen der Business-Logik um die CRUD-Methoden der Bundesländer und Bezirke erweitert bzw. die Funktionen mit Zeitfenstern entfernt.

### 3. Inbetriebnahme

Um das Programm in Betrieb nehmen zu können, muss zuerst die Datenbasis importiert werden. Die gesamte Datenstruktur, sowie alle Meldungen und zusätzlich notwendigen Daten können mithilfe der Datei „CREATE\_DATABASE.sql“ angelegt werden.

Hierfür wird in Visual Studio im SQL Server Object Explorer eine Query auf den zuständigen SQL-Express-Server angelegt und eine Datenbank namens „Covid“ erstellt (siehe erste Zeile im SQL-File).

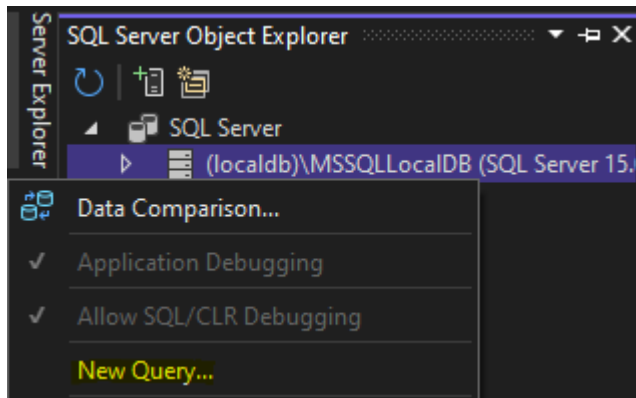


Abbildung 4: How-To Datenbank hinzufügen

Danach kann wiederum eine Verbindung/Query zu der neu erstellten Datenbank aufgebaut werden, um mit den restlichen Zeilen alle Daten anzulegen.

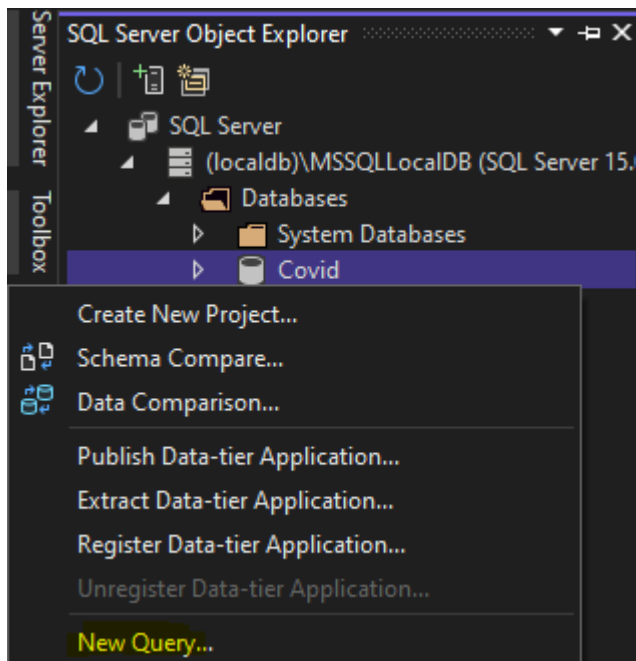


Abbildung 5: How-to: Daten zur Datenbank hinzufügen

## 4. Benutzerdokumentation

In diesem Bereich werden die wichtigsten Funktionen der Anwendung erläutert.

### 4.1. Übersicht

Beim Programmstart wird der Benutzer vom UI im „Übersicht“-Tab begrüßt. In diesem sind die momentan aktiven Fälle, die neuen Fälle für den heutigen Tag, sowie eine Übersicht über die aktuellen 7-Tage-Inzidenzen aller Bundesländer zu sehen. Diese Felder sind von den Einstellungen oben unbeeinflusst und werden nur zum Start des Programms berechnet.

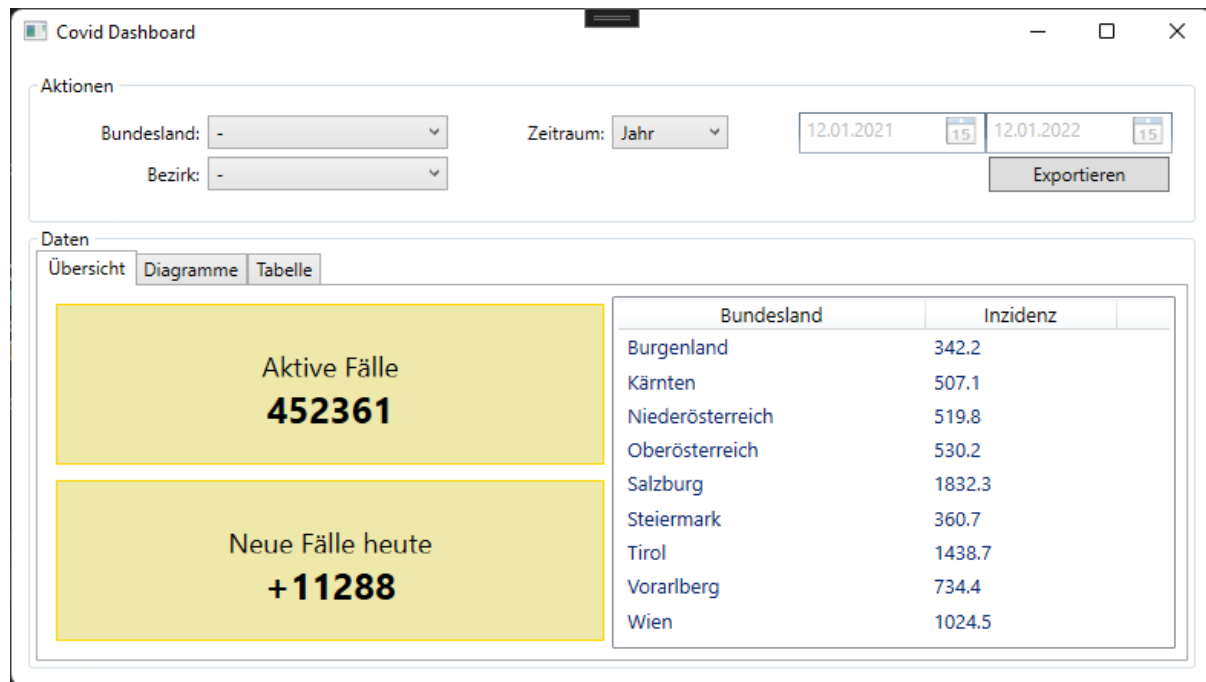


Abbildung 6: UI - Übersichts-Tab

Anmerkung für Tutoren:

(Stand 10.12.2022, neuester Datensatz)

Die momentanen aktiven Fälle fallen bei meiner Berechnung deutlich höher aus als erwartet (ist: 452361, soll: 77165). Im Endeffekt sollten sich die aktiven Fälle aus der Differenz zwischen Summe der Infizierten Personen und der Summe der Genesenen/Verstorbenen Personen ergeben. Also praktisch  $\text{Insgesamt\_Infiziert} - (\text{Insgesamt\_Genesen} + \text{Insgesamt\_Verstorben})$  – das scheint jedoch bei diesem Datenset nicht zu funktionieren.

Die neuen Fälle weichen leicht ab mit einem erwarteten Ergebnis von 10842 im Vergleich zu dem hier präsentierten Ergebnis von 11288.

#### 4.2. Filter

Oben in der Ansicht zu sehen, sind die Filter welche festgelegt werden können. Über Dropdowns kann eine Kombination von Bundesland, Bezirk und einem Datumsbereich ausgewählt werden, anhand dessen die Diagramme und die Tabelle neu berechnet werden.

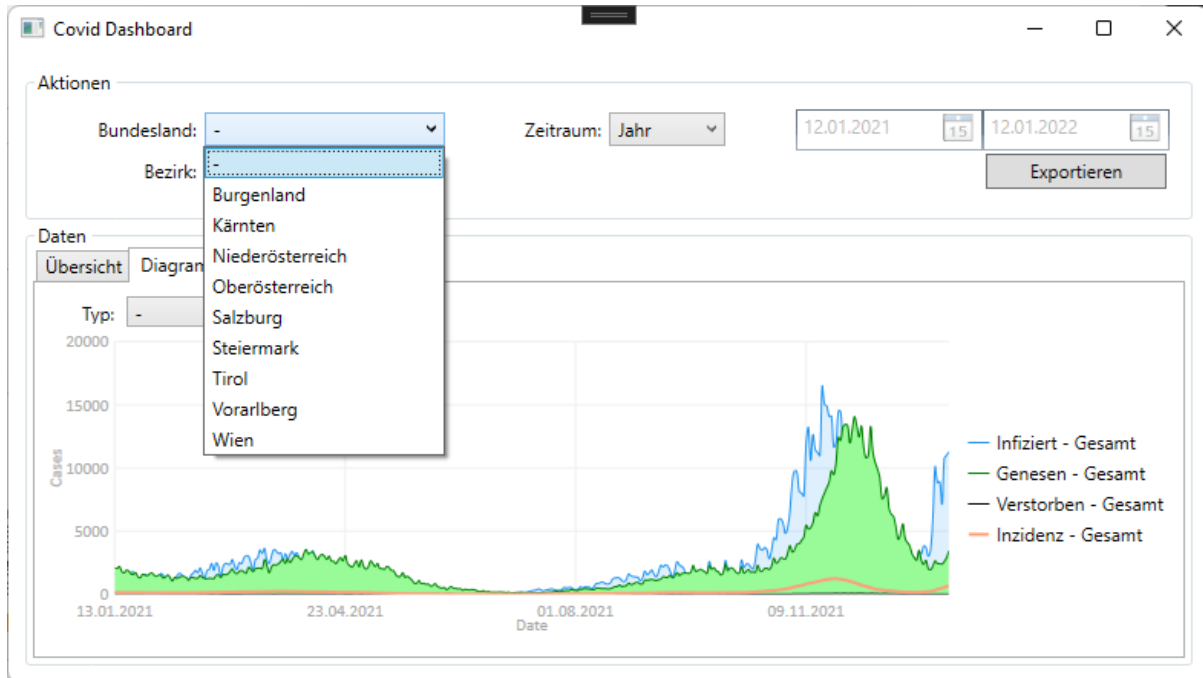


Abbildung 7: UI - Bundesland Dropdown

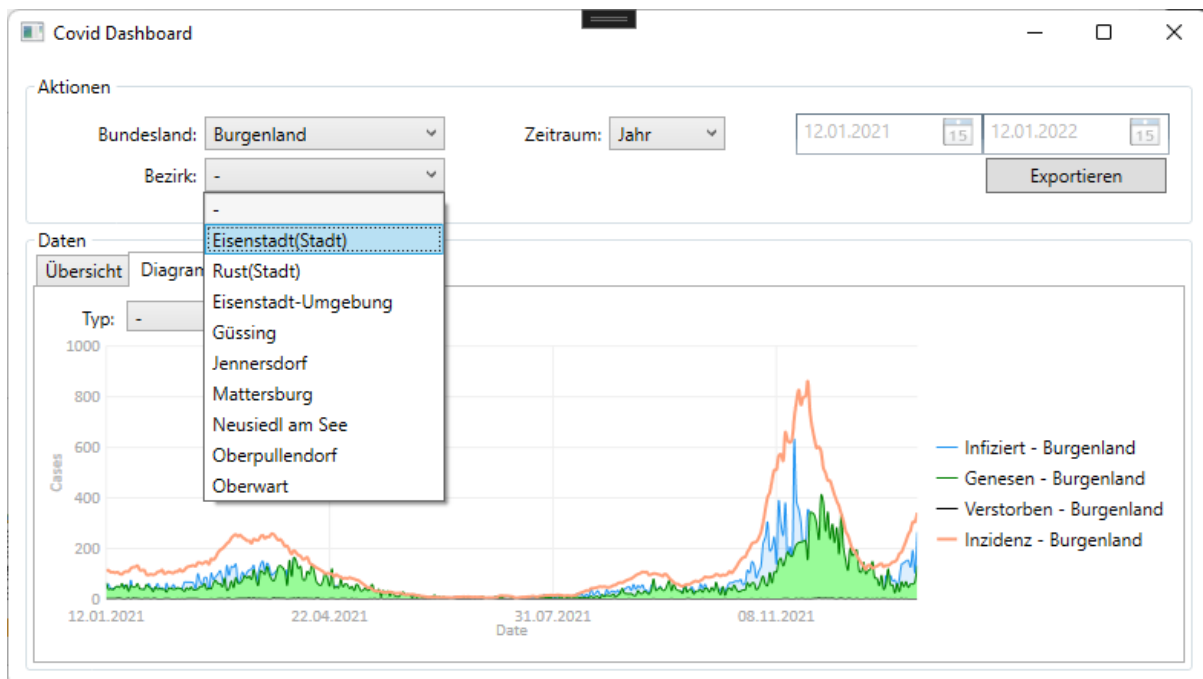


Abbildung 8: UI - Bezirk Dropdown



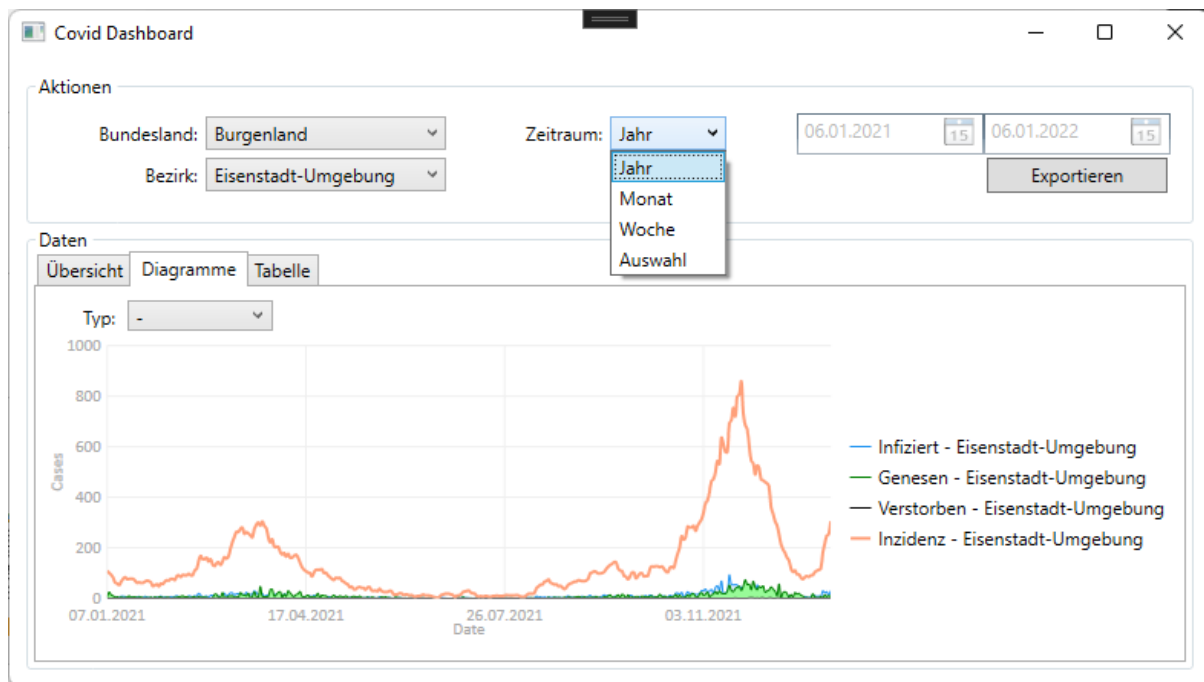


Abbildung 9: UI - Zeitraum Dropdown

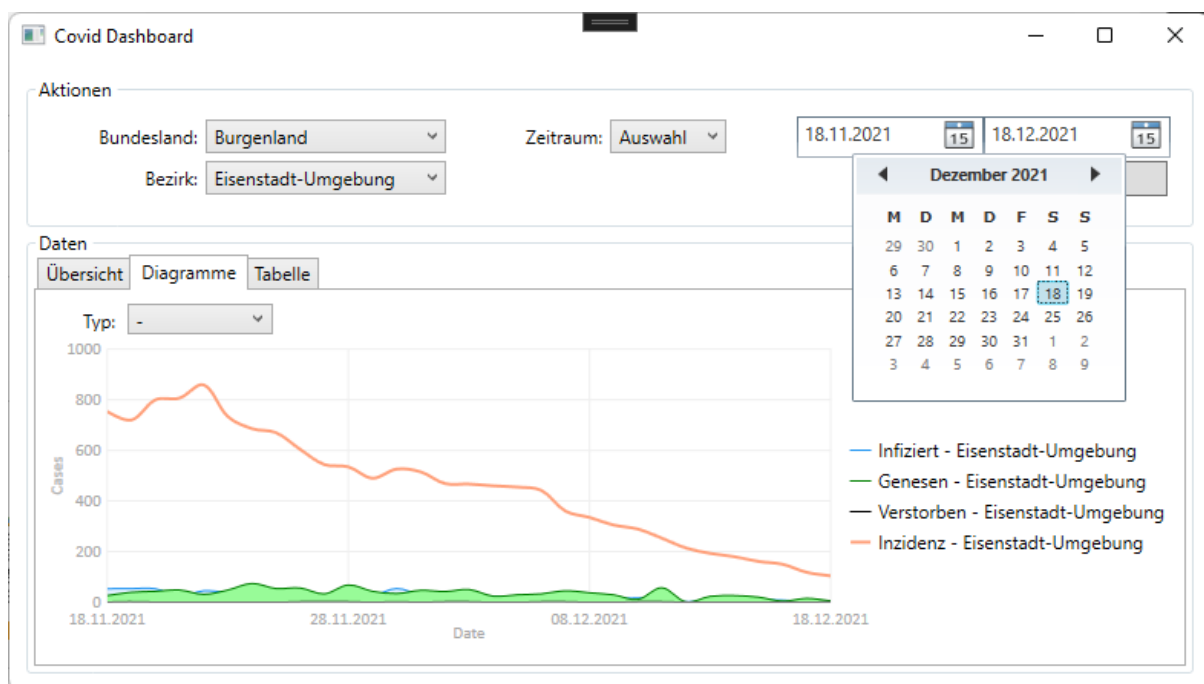


Abbildung 10: UI Datumsauswahl

### 4.3. Diagramm-Ansicht

Klickt man auf den Tab „Diagramme“, welcher schon während der Präsentation der Filter hergezeigt wurde, so erhält man einen Überblick über das Infektionsgeschehen in Österreich bzw. in den ausgewählten Regionen im gewählten Zeitraum.

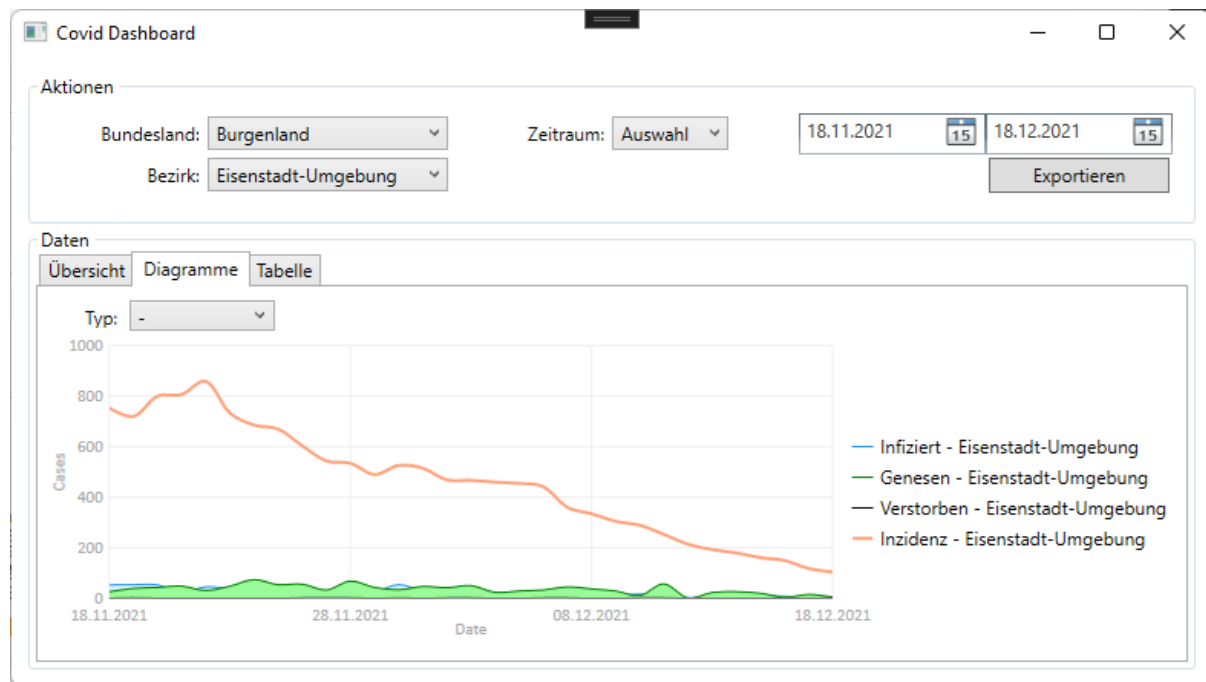


Abbildung 11: UI - Diagrammansicht - Gesamt

Standardmäßig werden alle Graphen gleichzeitig angezeigt, jedoch führt das zum Beispiel in diesem Szenario dazu, dass die Werte der Infektionen, Genesungen und Tode untergeht, da die Inzidenz zu hoch ist.

Aus diesem Grund, ist es möglich über die Combobox „Typ“ auszuwählen, welcher Graph dargestellt werden soll. Die Werte werden auch entsprechend für eine anständige Darstellung skaliert:

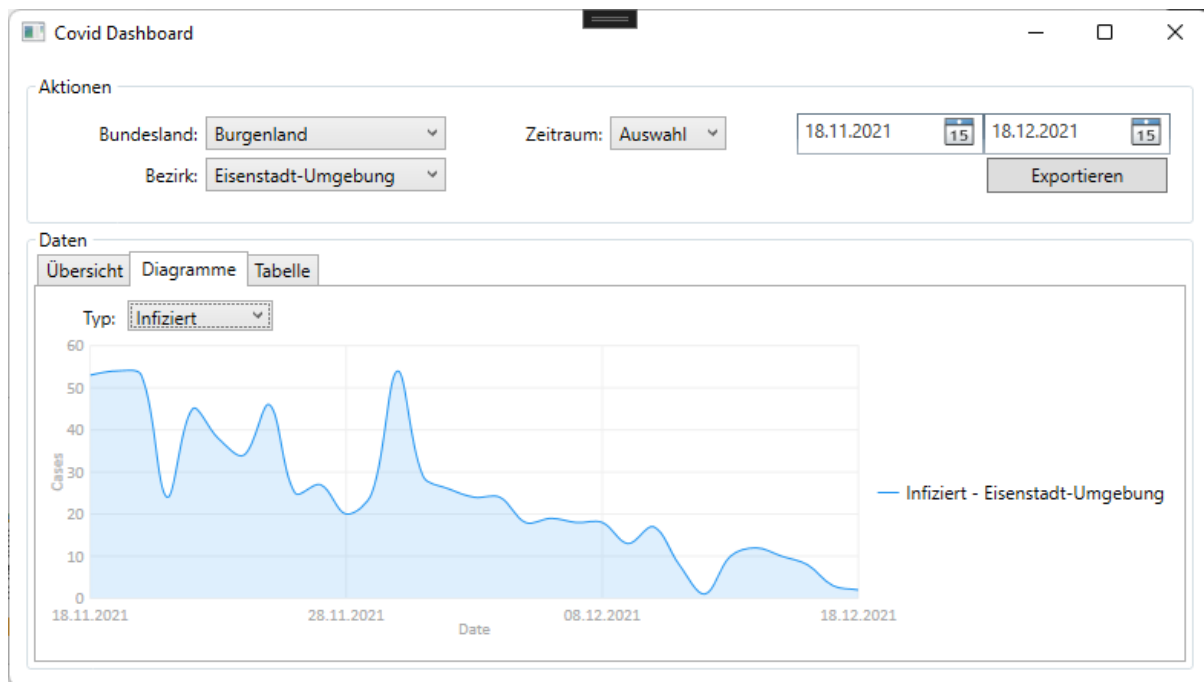


Abbildung 12: UI - Diagrammansicht - Infiziert

#### 4.4. Tabellen-Ansicht

Alternativ zum Diagramm können die Daten auch als Tabelle visualisiert werden.

The screenshot shows the 'Covid Dashboard' with the 'Tabelle' tab selected. The table displays the following data:

Datum	Bundesland	Bezirk	Inzidenz	Infiziert	Genesen	Gestorben
10.12.2021	Burgenland	Eisenstadt-Umgebung	289.55	17	11	1
11.12.2021	Burgenland	Eisenstadt-Umgebung	253.07	8	57	1
12.12.2021	Burgenland	Eisenstadt-Umgebung	214.31	1	1	0
13.12.2021	Burgenland	Eisenstadt-Umgebung	193.79	10	23	0
14.12.2021	Burgenland	Eisenstadt-Umgebung	180.11	12	26	0
15.12.2021	Burgenland	Eisenstadt-Umgebung	161.88	10	20	0
16.12.2021	Burgenland	Eisenstadt-Umgebung	150.48	8	4	0
17.12.2021	Burgenland	Eisenstadt-Umgebung	118.56	3	15	0
18.12.2021	Burgenland	Eisenstadt-Umgebung	104.88	2	5	1

Abbildung 13: UI – Tabellenansicht

#### 4.5. Export-Funktion

Über den Button „Export“ können Benutzer die dargestellte Tabelle als CSV exportieren. Aufgrund der Tatsache, dass der Decimal-Point im deutschen das Komma ist, wurde als Delimiter das Semicolon verwendet.

Für einen Export muss der Nutzer nur den Button „Exportieren“ anklicken und einen Speicherort festlegen.

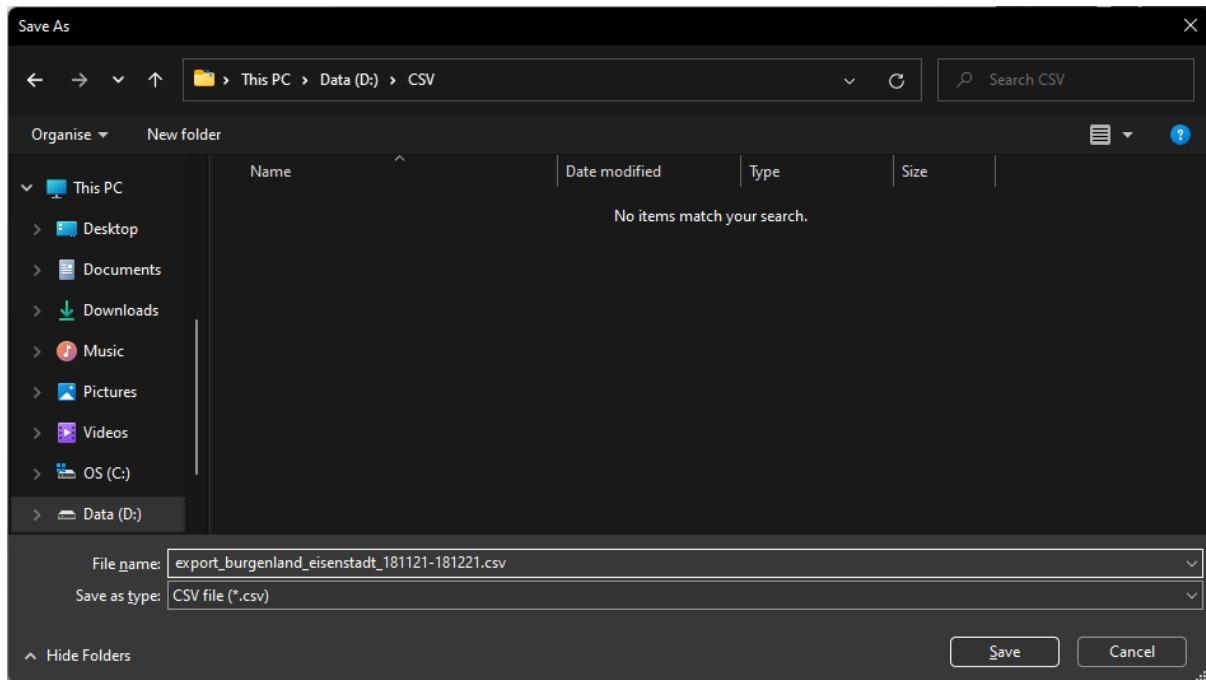


Abbildung 14: Export Dialog

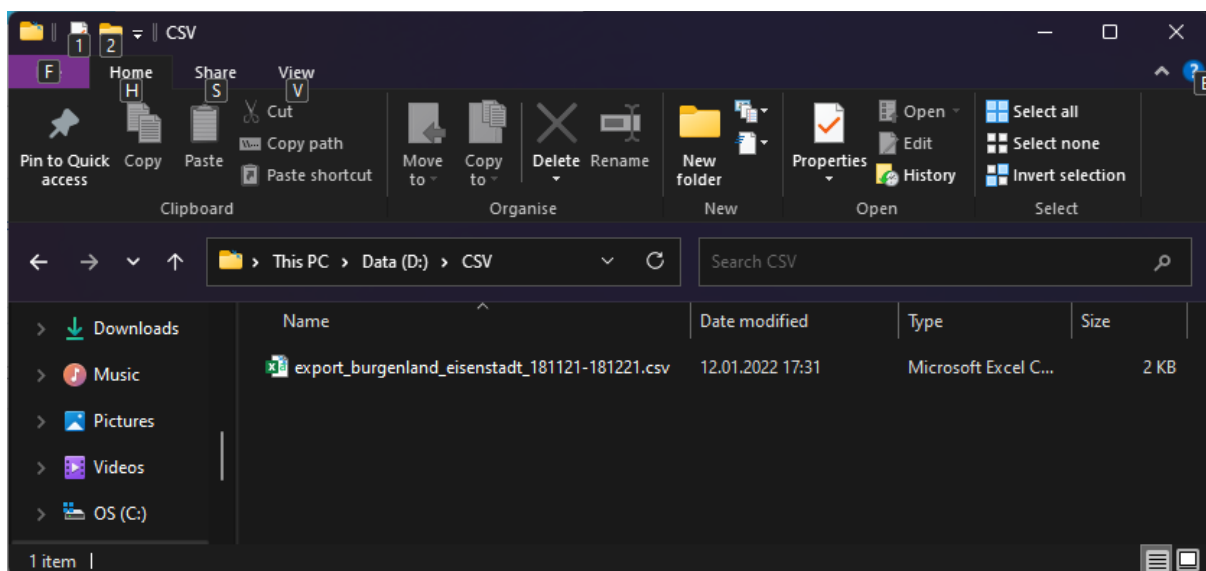
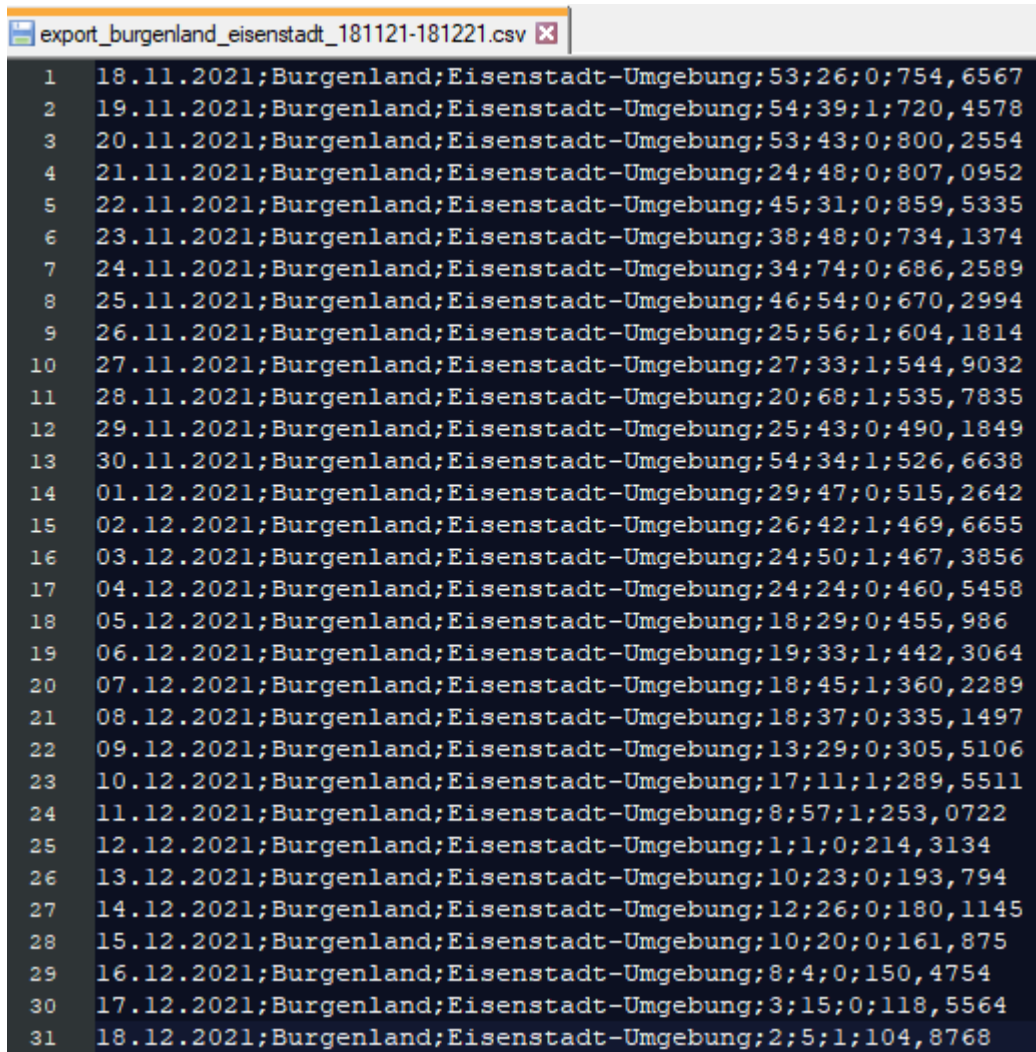


Abbildung 15: Export File



1	18.11.2021;Burgenland;Eisenstadt-Umgebung;53;26;0;754,6567
2	19.11.2021;Burgenland;Eisenstadt-Umgebung;54;39;1;720,4578
3	20.11.2021;Burgenland;Eisenstadt-Umgebung;53;43;0;800,2554
4	21.11.2021;Burgenland;Eisenstadt-Umgebung;24;48;0;807,0952
5	22.11.2021;Burgenland;Eisenstadt-Umgebung;45;31;0;859,5335
6	23.11.2021;Burgenland;Eisenstadt-Umgebung;38;48;0;734,1374
7	24.11.2021;Burgenland;Eisenstadt-Umgebung;34;74;0;686,2589
8	25.11.2021;Burgenland;Eisenstadt-Umgebung;46;54;0;670,2994
9	26.11.2021;Burgenland;Eisenstadt-Umgebung;25;56;1;604,1814
10	27.11.2021;Burgenland;Eisenstadt-Umgebung;27;33;1;544,9032
11	28.11.2021;Burgenland;Eisenstadt-Umgebung;20;68;1;535,7835
12	29.11.2021;Burgenland;Eisenstadt-Umgebung;25;43;0;490,1849
13	30.11.2021;Burgenland;Eisenstadt-Umgebung;54;34;1;526,6638
14	01.12.2021;Burgenland;Eisenstadt-Umgebung;29;47;0;515,2642
15	02.12.2021;Burgenland;Eisenstadt-Umgebung;26;42;1;469,6655
16	03.12.2021;Burgenland;Eisenstadt-Umgebung;24;50;1;467,3856
17	04.12.2021;Burgenland;Eisenstadt-Umgebung;24;24;0;460,5458
18	05.12.2021;Burgenland;Eisenstadt-Umgebung;18;29;0;455,986
19	06.12.2021;Burgenland;Eisenstadt-Umgebung;19;33;1;442,3064
20	07.12.2021;Burgenland;Eisenstadt-Umgebung;18;45;1;360,2289
21	08.12.2021;Burgenland;Eisenstadt-Umgebung;18;37;0;335,1497
22	09.12.2021;Burgenland;Eisenstadt-Umgebung;13;29;0;305,5106
23	10.12.2021;Burgenland;Eisenstadt-Umgebung;17;11;1;289,5511
24	11.12.2021;Burgenland;Eisenstadt-Umgebung;8;57;1;253,0722
25	12.12.2021;Burgenland;Eisenstadt-Umgebung;1;1;0;214,3134
26	13.12.2021;Burgenland;Eisenstadt-Umgebung;10;23;0;193,794
27	14.12.2021;Burgenland;Eisenstadt-Umgebung;12;26;0;180,1145
28	15.12.2021;Burgenland;Eisenstadt-Umgebung;10;20;0;161,875
29	16.12.2021;Burgenland;Eisenstadt-Umgebung;8;4;0;150,4754
30	17.12.2021;Burgenland;Eisenstadt-Umgebung;3;15;0;118,5564
31	18.12.2021;Burgenland;Eisenstadt-Umgebung;2;5;1;104,8768

Abbildung 16: Export Ergebnis