



Abschlussprüfung Sommer 2021

Fachinformatiker für Anwendungsentwicklung

Dokumentation zur betrieblichen Projektarbeit

Veltins Dashboard

Web-Anwendung für Veltins Gewinnspiel-Statistiken

Prüfungsbewerber:

Ömer S. ACAR
Xxxxxx XX
XXXXXX XXXXXXXX

Umschulungsträger:



IT-Akademie Dr. Heuer GmbH
Konrad-Zuse-Straße 2b
44801 Bochum

Ausbildungsbetrieb:



Schleier-IT
Michaelstr. 24A
45138 Essen

Inhaltsverzeichnis

Tabellenverzeichnis	ii
Abbildungsverzeichnis	ii
Abkürzungsverzeichnis	ii
1 Einleitung	1
1.1 Projektbeschreibung	1
1.2 Projektziel	1
1.3 Projektbegründung	2
1.4 Projektschnittstellen	2
1.5 Projektumfeld	2
1.5.1 Ausbildungsbetrieb	2
1.5.2 Kunde	2
2 Planung	3
2.1 Projektphasen	3
2.2 Ressourcenplanung	3
2.3 Entwicklungsprozess	4
2.4 Ist-Analyse	4
2.5 Soll-Konzept	4
2.6 Pflichtenheft	5
2.7 Wirtschaftlichkeitsanalyse	5
2.7.1 „Make or Buy“-Entscheidung	5
2.7.2 Projektkosten	6
2.7.3 Amortisationsdauer	6
2.8 Programmablaufplan	7
2.9 Entwicklungsumgebung und die Bibliotheken	7
3 Implementation	8
3.1 Benutzeroberflächen	9
3.2 Datenbankkommunikation	11
3.3 Geschäftslogik	12
4 Test/Qualitätssicherung	13
4.1 Genauigkeit	13
4.2 Benutzerfreundlichkeit & Verständlichkeit	13
4.3 Ausnahmebehandlung	14
5 Dokumentation	14
6 Fazit	14
6.1 Ist-Soll Vergleich	14
6.2 Gewonnene Erkenntnisse	15
Literaturverzeichnis	a

Anhang	b
A.1 Detaillierte Zeitplanung	b
A.2 Verwendete Ressourcen	c
A.3 Ausschnitt des Kanban-Boards	c
A.4 Auszug aus dem Pflichtenheft	d
A.5 Beispiel Screenshots der Programmablaufplan	e
A.6 Ein Screenshot der Benutzeroberfläche	f

Tabellenverzeichnis

Tabelle 1 - Grobe Zeitplanung	3
Tabelle 2 - Berechnung der Projektkosten	6
Tabelle 3 - Soll/Ist Vergleich	14

Abbildungsverzeichnis

Abbildung 1 - Aktionsauswahl	9
Abbildung 2 - Gesamtzahlen einer Aktion	9
Abbildung 3 - Zeitfilterwerkzeugen	9
Abbildung 4 - Aktivitätstabelle	10
Abbildung 5 – Geschlechter - Alter Verhältnis, Herkunft-Tabelle	10
Abbildung 6 - Aktionsvergleich	11

Abkürzungsverzeichnis

AJAX	Asynchronous JavaScript and XML
CSS	Cascading Style Sheets
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
MVC	Model-View-Control
NPM	Node Package Management
PHP	Hypertext Preprocessor
XML	Extensible Markup Language

1 Einleitung

Die folgende Projektdokumentation schildert und erläutert den Verlauf des IHK-Abschlussprojektes, welches der Autor im Rahmen seiner Ausbildung zum Fachinformatiker in der Fachrichtung Anwendungsentwicklung durchgeführt hat. Absicht dieser Projektdokumentation ist die Erläuterung des Prozesses von der Planung bis hin zum Einsatz der Webanwendung.

1.1 Projektbeschreibung

Das Unternehmen Veltins organisiert verschiedene Gewinnspiele für Werbezwecke, um seine Produktivität zu erhöhen. Die Bewertung und Interpretation dieser Gewinnspiele ist wichtig für die Verkaufs- und Werbekampagnenrichtlinien des Unternehmens. Durch Auswertung der Gewinnspiele kann der Kunden ein Wachstum der Teilnehmer Jahr für Jahr auswerten, Marketing-Aktionen wie Facebook-Postings, Online-Werbekampagnen oder Push-Benachrichtigungen direkt mit der Zahl und der Eigenschaften neuer Teilnehmer in Gewinnspielen korrelieren lassen und so die Aktionen bewerten.

In diesem Projekt geht es darum ein webbasiertes Dashboard zu entwickeln, welche dafür genutzt wird, alle Statistiken und Berechnungen eines Gewinnspiels über verschiedene Funktions- und Kategorieoptionen zu erstellen und mit Grafiken zu präsentieren. Die hierfür erforderlichen Daten werden aus der vorhandenen Gewinnspiel-Datenbank entnommen.

Da es sich um einen direkten Kundenauftrag handelt, wird die genaue Wirtschaftlichkeit des Projekts in der Dokumentation dargestellt. Die Planung des Projekts wird so angelegt, dass die Durchführung im Rahmen des vereinbarten Budgets bleibt und für die Schleier-IT ein positives wirtschaftliches Ergebnis liefert.

1.2 Projektziel

Ziel des Projekts ist die Entwicklung einer Webanwendung mit grafischer Benutzeroberfläche, Datenbankverbindung und Geschäftslogik. Auf diese Weise wird die Erstellung von Statistiken und Berechnungen für diese Wettbewerbe automatisiert, genauer, effizienter und aktueller bewertet und die für diesen Prozess erforderlichen Kosten reduziert.

1.3 Projektbegründung

Die manuelle Durchführung dieser Auswertungen ist sehr Zeit- und Arbeitsaufwendig. Darüber hinaus sind die Bewertungen aufgrund der ständigen Änderung der vorhandenen Gewinnspielsdaten schnell veraltet. Daher benötigte der Kunde dieses Dashboard, in dem diese Statistiken erstellt, wie gewünscht dargestellt und gleichzeitig aktualisiert werden.

1.4 Projektschnittstellen

Damit diese Anwendung funktioniert, ist Zugriff auf eine MySQL-Datenbank erforderlich. Diese Datenbank ist auf dem Server verfügbar, auf dem die Anwendung veröffentlicht wird. Abgesehen davon ist das Projekt unabhängig von allen anderen technischen Schnittstellen.

1.5 Projektumfeld

Die Projektumfeld wird in zwei Teilen als Ausbildungsbetrieb und Kunde bewertet.

1.5.1 Ausbildungsbetrieb

Der Ausbildungsbetrieb ist Schleier-IT, ein IT-Dienstleister, der von Kunden aus verschiedenen Branchen in Auftrag gegeben werden und seinen Kunden mit seiner 15 Mitarbeitern seit 2014 Webdesign, Software- und mobile App-Entwicklung, Webhosting und Domains anbietet. Das Projekt wird für einen Kunden von Schleier-IT eingesetzt werden und bei Schleier IT durchgeführt.

1.5.2 Kunde

Veltins GmbH & Co. KG ist eine im Jahre 1824 gegründete Brauerei und hat mehr als 600 Mitarbeitern. Der Sitz befindet sich in Meschede-Grevenstein im Sauerland. Schleier-IT betreut das Unternehmen Veltins in verschiedenen Bereichen, speziell die Veltins App. Die Gewinnspiele werden derzeit innerhalb der App abgewickelt. Schleier-IT wird den Serverdienst dieser Anwendung bereitstellen.

2 Planung

In diesem Abschnitt geht es darum, wie das weitere Vorgehen bestimmt wurde. Das Projekt wurde dazu zeitlich geplant, alle benötigten Ressourcen ermittelt, der Entwicklungs- und Testprozesse bestimmt.

2.1 Projektphasen

Der Autor stand ein Durchführungszeitraum von 70 Stunden zur Verfügung. Eine grobe Zeitplanung findet sich in Tabelle 1 - Grobe Zeitplanung. Eine detailliertere Übersicht mit den einzelnen Teilaufgaben kann im Anhang A.1 Detaillierte Zeitplanung eingesehen werden. Die Projektphasen basieren auf dem Wasserfallmodell, das ein lineares Vorgehensmodell ist und das in aufeinander folgenden Projektphasen organisiert ist. Der Grund für die Wahl des Wasserfallmodells besteht darin, sicherzustellen, dass der Arbeitsumfang und die Kosten zu Beginn des Projekts gut geschätzt werden können.

Projektphase	Geplante Zeit
A - Analyse & Planung	19h
B - Implementation	32h
C - Test/Qualitätssicherung	4h
D - Dokumentation	13h
E - Nachbearbeitung	2h
Gesamt	70h

Tabelle 1 - Grobe Zeitplanung

2.2 Ressourcenplanung

In der Übersicht, die sich im Anhang A.2 Verwendete Ressourcen befindet, sind jene Ressourcen aufgelistet, welche für dieses Projekt eingesetzt wurden. Hier wurde neben den Hard- und Softwareressourcen auch das Personal mit aufgenommen. Bei der Auswahl der Software wurde darauf geachtet, dass diese kostenfrei (z.B. Open source) zu der Verfügung steht oder Schleier-IT bereits über die passenden Lizenzen verfügt. Die Projektkosten sollen dadurch möglichst gering gehalten werden.

2.3 Entwicklungsprozess

Um eine flexible Umsetzung der Anforderungen zu ermöglichen, hat sich der Autor für einen agilen Entwicklungsprozess entschieden. Da es sich um ein vergleichsweise kleines Projekt mit wenigen Projektteilnehmern handelt, wurde die agile Methode Kanban verwendet. Die Methode Kanban ermöglicht die flexible Änderung der Anforderungen und einen fließenden Entwicklungsprozess, ohne dabei viele Einschränkungen zu definieren. Die Umsetzung erfolgt mit Hilfe eines Kanban-Boards, das in der Projektmanagement-Software "ClickUp" verwaltet wird. Die Projektphasen werden hier als Aufgaben hochgeladen, um den Entwicklungsprozess zu verfolgen. Ein Ausschnitt dieses Boards befindet sich im Anhang A.3 Ausschnitt des Kanban-Boards.

Die Anwendung wird zunächst mit Hilfe der Visual Studio als IDE und der Ampps-Anwendung, die den Apache-Server und die MySQL-Datenbank bereitstellt, auf dem lokalen Computer entwickelt und dann auf den Schleier-IT-Server übertragen. Bei der Entwicklung der Anwendung werden Beispieldatenbanken, Tabellen und Daten verwendet, die gemäß der tatsächlichen Datenstruktur erstellt werden.

2.4 Ist-Analyse

Die Auswertungen werden derzeit manuell für mehrere Tabellenkalkulationsdateien durchgeführt. Alle Teilnehmerdaten werden von einem Mitarbeiter von Schleier-IT mithilfe mehrerer SQL-Abfragen manuell aus verschiedenen Datenbanken als Tabellenkalkulationsdatei exportiert. Nach dem Export werden die Berechnungen dank der Tabellenkalkulationsfunktionen durchgeführt. Die berechneten Werte werden in eine Matrix in dieser Tabellenkalkulationsdatei eingegeben und mit manuell erstellten Grafiken angezeigt und ausgewertet. Dieser Prozess ist sehr zeit- und arbeitsintensiv und kostet den Kunden. Darüber hinaus ist das Tabellenkalkulationsformat für diese Prozesse sehr verwirrend und es müssen täglich neue Tabellen erstellt werden, da sich die Situation ständig ändert.

2.5 Soll-Konzept

Für dieses Projekt wird eine Webanwendung erstellt. Die Daten werden direkt aus der vorhandenen Veltins Gewinnspiel-Datenbank abgerufen, die bereits von Schleier-IT administriert wird. Schleier-IT wird den Serverdienst dieser Anwendung bereitstellen. Der

Zugriff auf die Anwendung erfolgt über ein Anmeldesystem durch Eingabe eines Benutzernamens und eines Kennworts. Statistiken, Auswertungen und Berechnungen variieren dynamisch mit Optionen und Eingabewerkzeugen, die dem Benutzer auf der Benutzeroberfläche zur Verfügung stehen.

Entsprechend den Kundenanforderungen sollte das Dashboard auch in der Lage sein, zwischen den Aktionen zu wechseln, um bessere Statistiken über die Teilnehmer während und nach der Durchführung von Gewinnspielen über Zeitoptionen wie heute, gestern, aktuelle Woche, letzte Woche oder ein beliebiges Zeitintervall zu erhalten. die Teilnehmer nach Geschlecht, Altersgruppe, Region und Zeitpunkt der Teilnahme zu bewerten und die verschiedenen Gewinnspiele mit den Vorjahren vergleichen zu können.

Darüber hinaus kann es zwischen Bundesländern wechseln und eine detaillierte regionale Bewertung der Teilnahme an einer Aktion anzeigen. Die aktuellsten Statistiken können jedes Mal abgerufen werden, wenn die Anwendung geöffnet, die Seite aktualisiert oder eine Funktion verwendet wird.

2.6 Pflichtenheft

Das Pflichtenheft legt die in der Entwicklung verfolgten Ziele des Produktes fest. Es wird verwendet, um den Entscheidungsraum für die Realisierung abzustecken. Diese Abgrenzung erfolgt durch das Festlegen von Wunsch-, Muss- und Absteckungskriterien. Ein Auszug aus dem für dieses Projekt erstellten Pflichtenheft befindet sich im Anhang A.4 Auszug aus dem Pflichtenheft.

2.7 Wirtschaftlichkeitsanalyse

Aufgrund des geschilderten Sachverhalts, der in Abschnitt 1.4 Projektbegründung und in Abschnitt 2.4 Ist-Analyse beschrieben wurde, ist die Umsetzung des Projekts erforderlich. Ob die Realisierung und die damit verbundenen wirtschaftlichen Aufwendungen gerechtfertigt sind, soll in den folgenden Abschnitten betrachtet werden.

2.7.1 „Make or Buy“-Entscheidung

Dieses Dashboard ist ein unternehmensspezifisches Produkt der Veltins. Bei der Recherche und Betrachtung von Software mit vergleichbarer Funktionalität konnte festgestellt werden,

dass keine der vorhandenen Anwendungen am Markt die Anforderungen erfüllt und im Verhältnis zu den im nächsten Abschnitt kalkulierten Projektkosten steht. Daher soll das Projekt in Eigenentwicklung durchgeführt werden.

2.7.2 Projektkosten

Im Folgenden sind die berechneten Kosten aufgeführt, die während der Entwicklung des Projekts entstanden sind. Dabei werden sowohl die Personal-, als auch sonstige Ressourcenkosten berücksichtigt. Als Ressourcenkosten wurde vom Management eine Pauschale von 10,00 € pro Stunde festgelegt, die sich aus mehreren Komponenten zusammensetzt. Diese Komponenten umfassen Stromkosten, Büromietkosten, Anschaffungskosten für Hardware und Software, Wartungs- und Lizenzkosten für den Server sowie Gemeinkosten. Da die exakten Personalkosten aus Datenschutzgründen nicht herausgegeben werden dürfen, wurde die Kalkulation anhand von beispielhaften Kosten durchgeführt. Ein beispielhafte Stundenpauschale eines Mitarbeiters liegt bei 50,00 €, die eines Auszubildenden 10,00 €. Sämtliche anfallende Projektkosten, können der folgenden Tabelle entnommen werden.

Vorgang	Mitarbeiter	Zeit	Personalkosten	Ressourcenkosten	Gesamtkosten
Entwicklung	Auszubildender	70h	700,00 €	700,00 €	1.400,00 €
Code-Review	Ausbilder	2h	100,00 €	20,00 €	120,00 €
Abnahme	Ausbilder	1h	50,00 €	10,00 €	60,00 €
					1.580,00 €

Tabelle 2 - Berechnung der Projektkosten

2.7.3 Amortisationsdauer

Im Folgenden soll geprüft werden, ab welchem Zeitpunkt sich das Projekt amortisiert. Ein Mitarbeiter der Schleier-IT sollte täglich diese Berechnungen machen und das Ergebnis dem Kunden schicken. Ein Mitarbeiter dem Kunden sollte die Statistiken und Bewertungen erstellen. Dafür müssen beide Mitarbeiter etwa 30 Minuten am Tag arbeiten. Dies erfordert insgesamt 1 Stunde Arbeit pro Tag. Oben wurde für einen Mitarbeiter eine beispielhafte Stundenpauschale von 50 € pro Stunde angegeben.

Demzufolge;

$1 \text{ Std.} \times 50 \text{ €} / \text{Std.} = 50 \text{ € Tageskosten}$

$1580 \text{ €} / 50 \text{ €} = 31,6 \sim 32 \text{ Tage}$

Es gibt ungefähr 20 Arbeitstage pro Monat;

$32 \text{ Tage} / 20 \text{ Arbeitstage} = 1,6 \text{ Monat}$

Nach 32 Arbeitstage oder 1,6 Monat wird sich das Projekt amortisieren.

2.8 Programmablaufplan

Für jede Aufgabe, die zur Implementierung der Anwendung erforderlich ist, wird ein Programmablaufplan erstellt. Es beschreibt den Algorithmus und die Abfolge der Operationen im Voraus, um die Aufgaben zu lösen. Das Programmablaufplan für dieses Projekt wird mit der PapDesigner-Anwendung erstellt. Beispiel Screenshots des Programmablaufplans findet sich im Anhang unter A.5 Beispiel Screenshots des Programmablaufplans.

2.9 Entwicklungsumgebung und die Bibliotheken

In diesem Abschnitt hat der Autor die folgenden Entscheidungen über die Entwicklungsumgebung und die Bibliotheken getroffen, die als Ergebnis seiner Forschung verwendet werden sollen.

Als Entwicklungsumgebung wird Visual Studio Code verwendet, da es das Erstellen von Websites mit HTML, CSS, JSON, PHP und JavaScript unterstützt und Syntaxhervorhebung, automatische Vervollständigung, eckige Klammeranpassung usw. für Programmiersprachen bietet¹.

Da es Open Source und kostenlos ist und von fast allen Servern unterstützt wird, wurde beschlossen, die Programmiersprache PHP² für die Kommunikation mit der vorhandenen MySQL-Datenbank zu verwenden.

¹ Timotic, 2018

² Thattil, 2015

NPM wird als Paketmanager für Frontend verwendet, da es über die größte Bibliothek verfügt und dem Entwickler den Komfort bietet³, die erforderlichen Module einfach zu laden. Als andere Paketmanager wird Composer verwendet. Composer ist ein anwendungsorientierter Paketmanager für die Skriptsprache PHP⁴.

Obwohl die Teilnehmerdatenstruktur keine geschlechtsspezifischen Informationen enthält, fordert der Kunde auch eine Bewertung der Teilnehmer nach Geschlecht an. Der Autor suchte nach einer Lösung, die eine hohe Genauigkeit bietet und kostenlos ist. Hierzu wird das Gender-Detector-Paket⁵ verwendet, das mit Hilfe von Composer in das Projekt aufgenommen wird. In diesem Paket wird das Geschlecht des Teilnehmers anhand seines Vornamens und seines Landes geschätzt.

Verschiedene Bibliotheken wie Chart.js, C3.js, Chartist.js, MetricsGraphics.js wurden untersucht und getestet, um zu entscheiden, welches JavaScript-Framework für Statistiken, Diagramme und Grafiken im Dashboard verwendet wird. Chart.js wurde für die Datenvisualisierung entschieden, weil es visuelle und funktionale Merkmale nach Kundenwunsch und eine kurze und einfache Syntax bietet, die ein einfaches Debuggen und Aktualisieren ermöglicht. Die Moment.js-Bibliothek wird für die Zeitrahmenauswahl verwendet, und Bootstrap, eine CSS-Bibliothek, die mit ihren vordefinierten Klassen für die Benutzeroberfläche Komfort bietet, wird neben HTML und CSS auch verwendet.

3 Implementation

In diesem Abschnitt wird das Projekt in Übereinstimmung mit Pflichtenheft, Zeit- und Ressourcenplanung, ausgewählten Entwicklungswerkzeugen und Ablaufplan durchgeführt. Dieses Projekt wird nach dem MVC-Modell entworfen. Diese Vorgehensweise bietet eine geordnete Architektur, wodurch die Möglichkeit ausgeschlossen wird, mit dem falschen Element umzugehen oder einige Elemente zu vergessen. Diese Vorgehensweise bietet auch eine klare Unterscheidung zwischen Benutzeroberfläche, Modellen und Logik. Bei dieser Methode werden zuerst die erforderlichen Klassen und Prozeduren erstellt. Durch Erstellen von Objekten aus diesen Klassen wird dann die Logik der Daten erstellt, die in die Ansicht übertragen werden sollen. dann werden die erhaltenen Daten in der Ansicht angezeigt.

³ Rhymes, 2018

⁴ Wikipedia, 2020

⁵ Michael, 2020

3.1 Benutzeroberflächen

In dieser Phase wird die Benutzeroberfläche (View) erstellt und Methoden, die in der vorherigen Phase erstellt wurden, mittels Tools wie Dropdown-Menü und Schaltflächen aufgerufen und die Ausgabe von Methoden durch die entsprechende Darstellung angezeigt. Entsprechend der ausgewählten Aktion wird eine Datenanforderung aus der entsprechenden Tabelle der Datenbank gestellt und die Berechnungsergebnisse für die ausgewählte Aktion werden angezeigt.



Abbildung 1 - Aktionsauswahl

Entsprechend der ausgewählten Aktion wird zuerst die Gesamtzahl der Teilnehmer für die gesamte Aktion, die Gesamtzahl der Gerubbelte Felder für die gesamte Aktion und die Anzahl der Teilnehmer für den aktuellen Tag angezeigt.

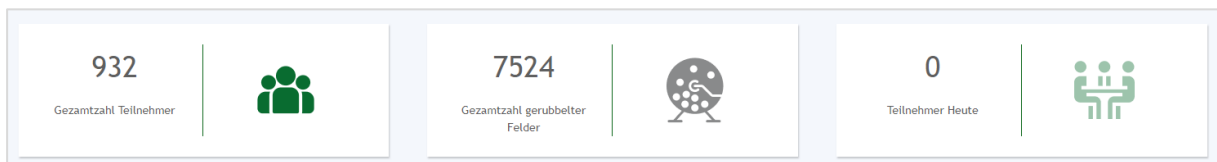


Abbildung 2 - Gesamtzahlen einer Aktion

Die ausgewählte Aktion wird zunächst standardmäßig als vollständige Aktion geladen. Anschließend können mithilfe von Zeitfilterwerkzeugen die Daten der Aktion in einem bestimmten Zeitintervall gefiltert werden.

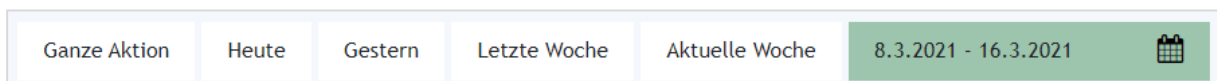


Abbildung 3 - Zeitfilterwerkzeugen

Die nächste Aktivitätstabelle zeigt Folgendes mit dem neuesten Eintrag oben: Täglicher Benutzer, tägliche neue Adressdaten, Anzahl der geriebenen Websites, kumulierte

Gesamtzahl der geriebenen Bereiche und Gesamtzahl der Teilnehmer. Diese Tabelle ändert sich dann dynamisch basierend auf der Auswahl von Heute, Gestern, Letzte Woche, Aktuelle Woche und einem beliebigen Datumsbereich.

Aktivität					
Datum	User pro Tag	Neue Adressdaten pro Tag	Anzahl gerubelter Felder	Gesamt gerubelter Felder	Gesamt Teilnehmer
13-03-2021	1	1	382	6839	1133
12-03-2021	1	1	333	6457	1132
11-03-2021	1	1	266	6124	1131
10-03-2021	1	1	215	5858	1130
09-03-2021	1	1	98	5643	1129

Abbildung 4 - Aktivitätstabelle

Anschließend wird das Geschlechterverhältnis der Teilnehmer für die gesamte Aktion in einem Donut-Diagramm und das Altersverteilungsverhältnis in einem Balkendiagramm angezeigt. In der Herkunft-Tabelle wird zunehmend die Verteilung der Teilnehmer nach Bundesländern für alle Aktionen angezeigt.

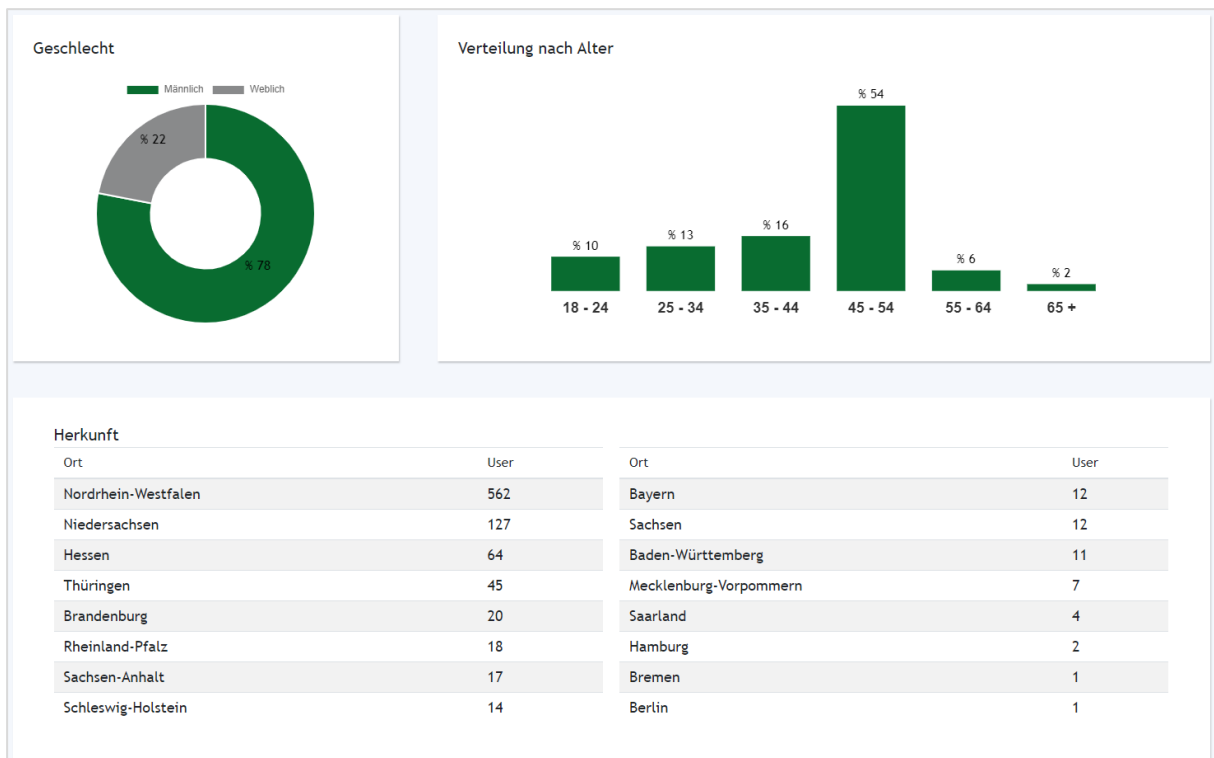


Abbildung 5 – Geschlechter - Alter Verhältnis, Herkunft-Tabelle

Am Ende des Dashboards befindet sich ein Abschnitt zum Aktionsvergleich. Alle Aktionen in der Datenbank werden Jahr für Jahr miteinander verglichen, indem die Zahlen in den beiden linearen Diagrammen von Benutzer und Gerubbelter Felder verglichen werden.

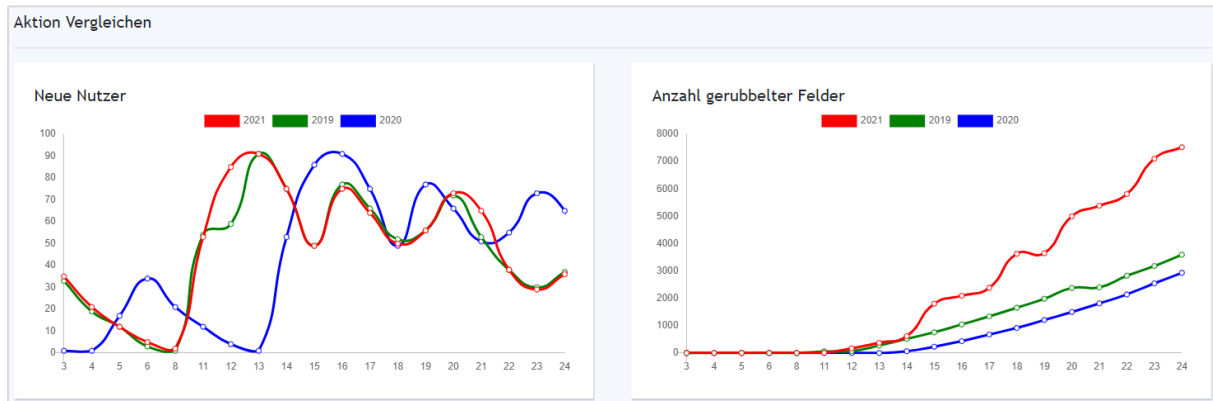


Abbildung 6 - Aktionsvergleich

Die Grundstruktur der Benutzeroberfläche wird mit HTML, CSS, Bootstrap, dynamische Abschnitte wie Tabelle werden mit der JavaScript Programmiersprache erstellt. Die Bibliothek JQuery und Moment.js wird für die Zeitintervallauswahlfunktion verwendet. Chart.js, eine kostenlose Open-Source-JavaScript-Bibliothek, wird für die Datenvisualisierung verwendet.

Das Design der Benutzeroberfläche wurde von einem Mediengestalter im Voraus mit Figma, einem webbasierten Vektorgrafik-Editor und Prototypen-Tool, gemäß den Kundenanforderungen vorbereitet. In Übereinstimmung mit diesem Design wird eine Benutzeroberfläche erstellt. Das verwendete Thema wird mit dem in anderen Veltins-Anwendungen verwendeten Thema kompatibel. Ein Screenshot der Benutzeroberfläche findet sich im Anhang unter A.6 Screenshots der Benutzeroberfläche.

3.2 Datenbankkommunikation

In dieser Phase wird die Verbindung der Dashboard-Anwendung mit MySQL-Datenbanken hergestellt. Die Struktur, die hier erstellt werden muss, sollte sowohl gegen SQL-Injektionen sicher als auch flexibel genug sein, um einen einfachen Zugriff auf mehr als eine Datenbank zu ermöglichen. Hierzu wird das MySQLi-Objekt von PHP verwendet, das als Datenbankabstraktionsschicht fungiert. MySQLi ist eine verbesserte Erweiterung von PHP zum Zugriff auf MySQL-Datenbanken und bietet die Freiheit mit mehr als einer Datenbank zu arbeiten.

Mit der PHP und MySQLi-Objekt wird eine Verbindungsklasse (Model) erstellt, um eine Verbindung zu Datenbanken auf dem MySQL-Server herzustellen. Für diese Klasse wird eine Methode erstellt, die den Datenbanknamen als Parameter verwendet, eine SQL-Abfrage an die Datenbank sendet und das Abfrageergebnis zurückgibt. In dieser Funktion wird die Geschlechtsbestimmung durchgeführt, die durch Erstellen eines Objekts der Gender-Detector-Klasse erfolgt.

Dann wird in einer anderen PHP-Datei (Control) ein Objekt dieser Linkklasse erstellt, die Methode der Klasse wird über das Objekt aufgerufen und das Ergebnis wird in das JSON-Format konvertiert. Damit die aktuelle Datenbank beim ersten Öffnen der Seite standardmäßig angezeigt wird, wird eine andere Methode erstellt, um die Namen der Datenbanken in MySQL Server abzufragen.

3.3 Geschäftslogik

Um beim Start standardmäßig die aktuelle Gewinnspieldaten zu laden, wird in der JavaScript-Datei (Control) eine Funktion erstellt, die beim erstmaligen Laden des Control Panels ausgeführt wird und eine AJAX-Abfrage an die entsprechende ausführt PHP-Datei.

Diese Funktion übernimmt Datenbanknamen aus der Datenbank und gibt die aktuelle Datenbank zurück. Es wird eine weitere Funktion geschrieben, die diesen zurückgegebenen aktuellen Datenbanknamen als Parameter verwendet. Diese Funktion ruft die erforderlichen Daten über eine andere AJAX-Abfrage aus der Datenbank ab und ermöglicht die Ausführung anderer Funktionen. Nachfolgende Datenanforderungen werden basierend auf der vom Benutzer gewählten Aktion gestellt.

Die Datenbanknamen von Gewinnspiel in MySQL Server werden mit einem Zeitstempel im Epochenformat am Anfang und einer benutzerdefinierten Erweiterung am Ende gespeichert. Dies geschieht, um die für das Dashboard erforderlichen Datenbanken von anderen zu unterscheiden und festzustellen, was aktuell ist. Die Anfrage wird unter Berücksichtigung dieser Anhänge gestellt.

Andere Funktionen zum Analysieren, Formatieren und Interpretieren von Daten werden erstellt, um die erforderlichen Datenobjekte für grafische Elemente zu erstellen. Mithilfe der

Chart.js-Bibliothek werden dann Objekte aus der Chart-Klasse erstellt und diese Datenobjekte verwendet, um die Visualisierung der Daten zu ermöglichen.

In dieser Phase werden auch die Methoden erstellt, die für einige benutzerfreundliche Oberflächenfunktionen erforderlich sind. Beispiel: Ändern der Farbe der aktiven Schaltfläche, Schreiben des ausgewählten Datumsbereichs als Text der Schaltfläche usw. JavaScript und eine freie JavaScript-Bibliothek jQuery wird für Frontend-Funktionen verwendet. JQuery bietet eine einfache Syntax für XMLHttpRequest, ein JavaScript-Objekt, das zum Übertragen von Daten über HTTP verwendet wird, und viele andere JS-Funktionen. Auf diese Weise können Inhalte asynchron hochgeladen werden.

Die Postleitzahlen der Teilnehmer sind in der Datenstruktur verfügbar. Das Dashboard sollte jedoch eine Tabelle enthalten, die die Verteilung der Teilnehmer nach Bundesländern zeigt. Zu diesem Zweck wurde eine Open Source JSON-Datei, die alle Postleitzahlen und Bundesländer für Deutschland enthält, aus dem Internet heruntergeladen und dem Projekt hinzugefügt. In dieser JSON-Datei wurden die Bundesländer der Teilnehmer durch Vergleich der Bundesländer und Postleitzahlen mit den Postleitzahlen in der Datenbank ermittelt und eine Verteilungstabelle nach Bundesländern erstellt.

4 Test/Qualitätssicherung

4.1 Genauigkeit

Mehrere Beispieldatenbanken und Daten wurden zum Testen der Anwendung erstellt. Es wurde getestet, ob die aus der Datenbank erhaltenen Daten berechnet und die korrekten Ergebnisse auf die Benutzeroberfläche übertragen wurden. Die Daten hierzu wurden mit der Console.log-Methode auf die Konsole gedruckt, manche Berechnungen wurden manuell durchgeführt und die Richtigkeit von den Berechnungen überprüft.

4.2 Benutzerfreundlichkeit & Verständlichkeit

Es wird sichergestellt, dass alle Schaltflächen, Menüs, Tabellen, Grafiken und Informationen auf der Benutzeroberfläche so gestaltet sind, dass sie für den Benutzer leicht verständlich sind, und dass informative Notizen oder Abbildungen verwendet werden. Zu diesem Zweck wurden die erforderlichen Erklärungen unter Verwendung des Attributs "title" in den HTML-Tags abgegeben, die aktiv sind, wenn der Benutzer den Mauszeiger über das Objekt bewegt.

4.3 Ausnahmebehandlung

Es wird sichergestellt, dass eine Fehlermeldung in der gesprochenen Sprache für alle Ausnahmen generiert wird, auf die der Benutzer möglicherweise stößt, einschließlich der Anmelde- und Abmeldeschritte. Dies sind Situationen, in denen der Benutzer keine Verbindung zum Internet herstellen kann, falsche Eingabe von Benutzername und Kennwort, Serverfehler, Probleme mit der Datenbankverbindung, die zu Beginn und während des Aktionswechsels auftreten.

5 Dokumentation

Die Projektdokumentation wurde als prozessorientierter Projektbericht fortwährend innerhalb des Projektes vom Autor manuell erstellt. Dieser beschreibt die einzelnen Phasen und deren Inhalte, die im Laufe des Projektes durchlaufen wurden.

6 Fazit

6.1 Ist-Soll Vergleich

Rückblickend wurden alle im Pflichtenheft definierten Anforderungen umgesetzt. Der zu Anfang erstellte Projektplan konnte ebenfalls eingehalten werden. In der Tabelle Soll/ Ist-Vergleich ist der benötigte Zeitaufwand gegenübergestellt. Es ist zu erkennen, dass es eine gewisse Zeitdifferenz bei der Implementationsphase gegeben hat. Da die Implementation-Phase des Dashboards mehr Zeit benötigte, konnte die Pufferzeit für diese Abweichung verwendet werden. Somit wurde das Projekt in den festgelegten 70 Stunden umgesetzt.

Projektphase	Geplante Zeit	Tatsächliche Zeit	Differenz
A - Analyse & Planung	20h	20h	0h
B - Implementation	32h	34h	+2h
C - Test/Qualitätssicherung	4h	4h	0h
D - Dokumentation	13h	13h	0h
E - Nachbearbeitung	2h	0h	-2h
Gesamt			70h

Tabelle 3 - Soll/Ist Vergleich

6.2 Gewonnene Erkenntnisse

Das Projekt ermöglichte es dem Autor, den Ablauf eines Softwareprojekts von Anfang bis Ende zu betrachten. Der wichtigste Faktor war nicht nur, in direkten Programmieraufgaben aktiv zu sein, sondern auch in erster Linie für die genaue Definition der technischen Anforderungen, die Auswahl des Entwicklungsprozesses und das Management der einzelnen Projektphasen verantwortlich zu sein. Darüber hinaus konnte der Autor wichtige Erfahrung im Bereich PHP-Programmierung und der MySQL-Datenbank sammeln.

Literaturverzeichnis

Michael, J. (31.12.2020). Gender Detector

<https://github.com/tuqqu/gender-detector>

Rhymes, C.S. (31.03.2018). What is NPM and why should I use it?

<https://medium.com/@chrisrhymes/what-is-npm-and-why-should-i-use-it-ef15de78e305>

Thattil, S. (19.03.2015). 13 Vorteile von PHP

<https://www.yuhiro.de/13-vorteile-von-php/>

Timotic, M. (05.11.2018). 14 Best Web Development IDE

<https://tms-outsource.com/blog/posts/web-development-ide>

Wikipedia, (03.11.2020) Composer (Paketverwaltung)

[https://de.wikipedia.org/wiki/Composer_\(Paketverwaltung\)](https://de.wikipedia.org/wiki/Composer_(Paketverwaltung))

Anhang

A.1 Detaillierte Zeitplanung

Projektphase	Geplante Zeit
A - Analyse & Planung	19h
Projektphasen und Ressourcenplanung	3h
Durchführung einer Ist-Analyse	2h
Erstellung eines Soll-Konzeptes	2h
Durchführung der Wirtschaftlichkeitsanalyse	3h
Erarbeitung eines Pflichtenheftes	4h
Erstellung des Programmstrukturplan	3h
Auswählen der Entwicklungsumgebung und die Bibliotheken	2h
B - Implementation	32h
Aufbau der Kommunikation mit Datenbanken	2h
Erstellung der SQL-Abfragen	2h
Erstellung der Elemente zur Datenspeicherung	1h
Konzipierung der Struktur von Benutzeroberflächen	3h
Erstellung der Logik für Login	3h
Erstellung der Logik für Aktivitäten	3h
Erstellung der Logik zur Auswertung von Geschlecht	3h
Erstellung der Logik zur Auswertung von Altersgruppen	3h
Erstellung grafischer Elemente	2h
Verbindung von Daten und grafischen Elemente	2h
Erstellung der Logik zum Wechsel von Aktionen	2h
Erstellung der Logik zum Wechsel von Datum	3h
Styling der Benutzeroberfläche	3h
C - Test/Qualitätssicherung	4h
Genauigkeit von Statistiken und Berechnungen	1h
Benutzerfreundlichkeit aller Funktionen	1h
Verständlichkeit von Tabellen und Grafiken	1h
Eine eindeutige Fehlermeldung für jeden möglichen Fehler	1h
D - Dokumentation	13h
E - Nachbearbeitung	2h
Gesamt	70h

A.2 Verwendete Ressourcen

Hardware

HP-Laptop	: Rechner
Samsung Bildschirm	: Zusätzlicher Bildschirm

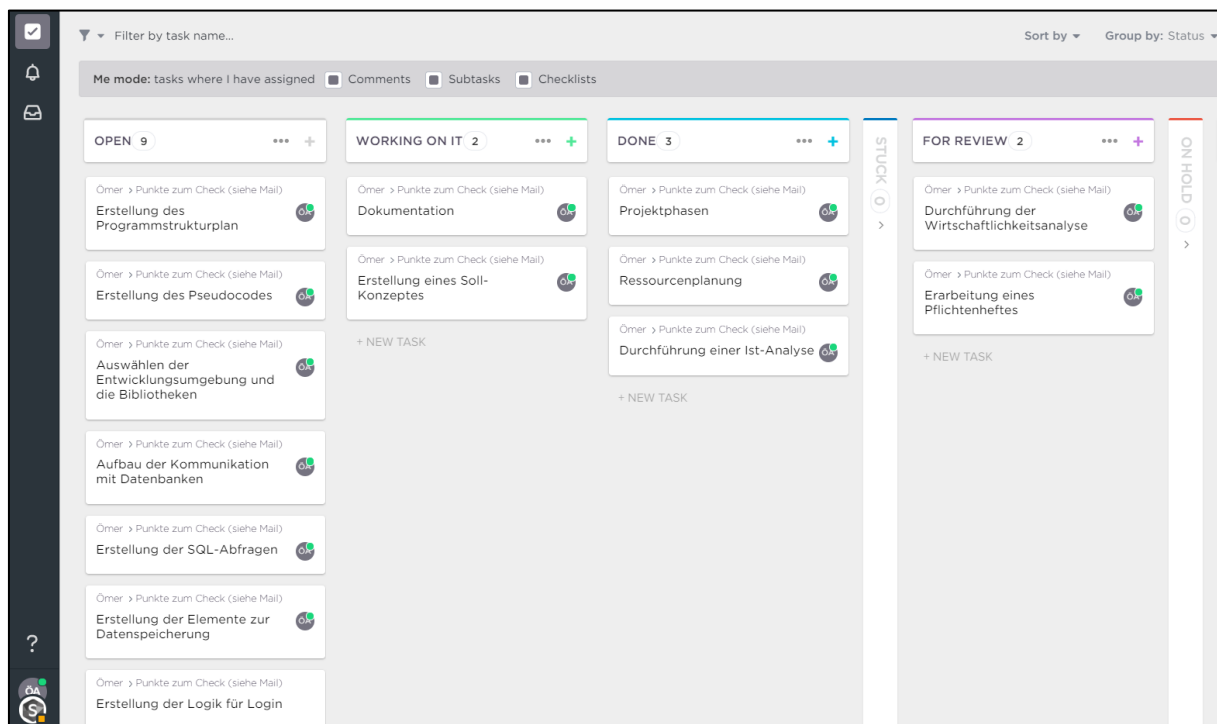
Software

Windows 10 Professional	: Betriebssystem
Visual Studio Code	: Entwicklungsumgebung
Git / GitHub	: Versionskontrolle
Npm	: Paketmanager für Frontend
Composer	: Paketmanager für PHP
Ampmps	: Apache Server, MySQL Datenbank, PHP 7.3
ClickUp	: Projektmanagement-Tool / Kanban-Board
Figma	: Designentwurf
Office 365	: Dokumentation
PapDesigner	: Programmablaufplan
Greenshot	: Screenshot

Personal

Anwendungsentwickler	: Umsetzung des Projektes
Ausbilder	: Review der Code und Abnahme

A.3 Ausschnitt des Kanban-Boards



The screenshot shows a Kanban board with the following columns and tasks:

- OPEN (9 tasks):**
 - Erstellung des Programmstrukturplan
 - Erstellung des Pseudocodes
 - Auswählen der Entwicklungsumgebung und die Bibliotheken
 - Aufbau der Kommunikation mit Datenbanken
 - Erstellung der SQL-Abfragen
 - Erstellung der Elemente zur Datenspeicherung
 - Erstellung der Logik für Login
- WORKING ON IT (2 tasks):**
 - Dokumentation
 - Erstellung eines Soll-Konzeptes
- DONE (3 tasks):**
 - Projektphasen
 - Ressourcenplanung
 - Durchführung einer Ist-Analyse
- FOR REVIEW (2 tasks):**
 - Durchführung der Wirtschaftlichkeitsanalyse
 - Erarbeitung eines Pflichtenheftes

Each task card includes a progress indicator (a green circle with a white dot) and a button labeled 'Punkte zum Check (siehe Mail)'.

A.4 Auszug aus dem Pflichtenheft

3 Beschreibung der Anforderungen

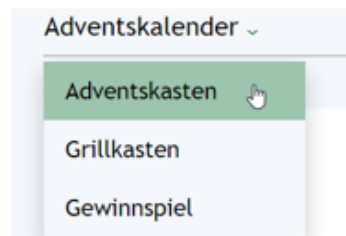
Für das Projekt besteht ein Design-Entwurf, welcher als Grundlage für die kommenden Funktionen gilt.

Link Designentwurf:

<https://www.figma.com/file/36qHU1q2TiBXTor9uBb5zm/Veltins?node-id=0%3A1>

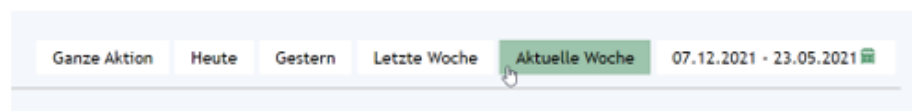
3.1 Wesentliche Teilfunktionen

a) Wechsel zwischen Aktionen



Es muss möglich sein, zwischen verschiedenen Aktionen und somit auch zwischen verschiedenen Datenbanken zu wechseln.

b) Datumsauswahl

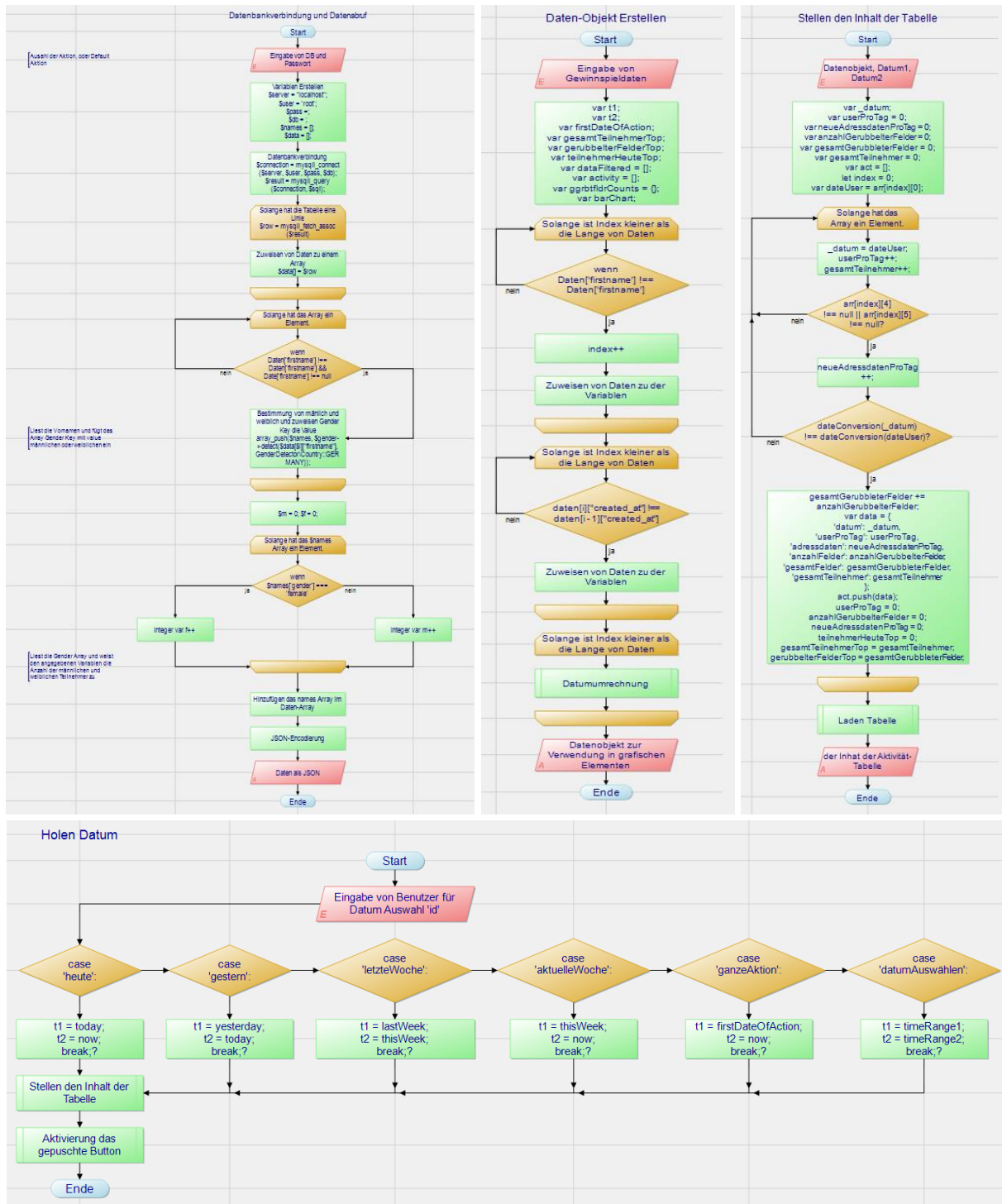


Es muss gewährleistet werden, dass auch individuelle Zeiträume innerhalb einer Aktion miteinander verglichen werden können.

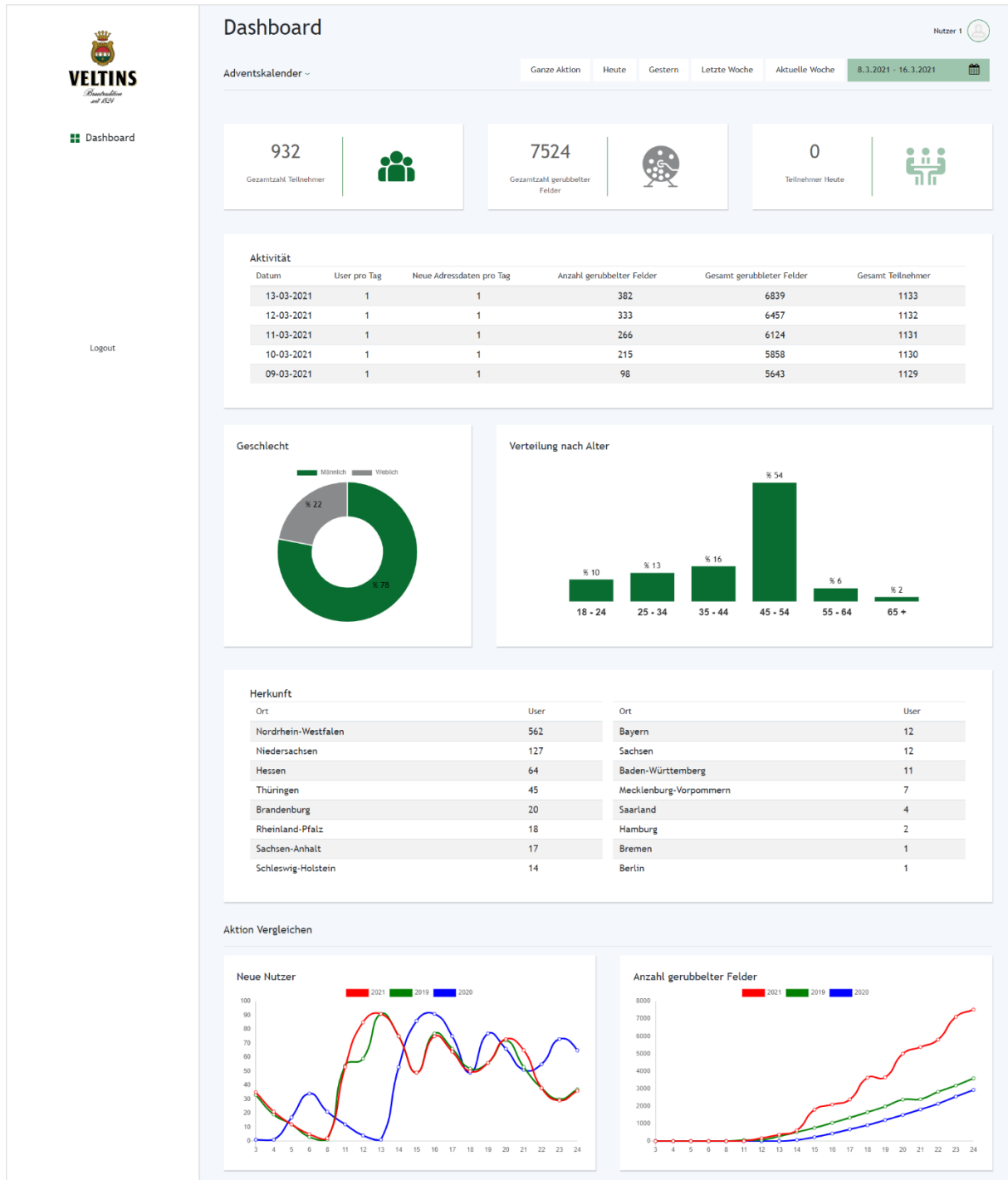
c) Modul Aktivitäten

Es soll möglich sein für jeden Tag die Anzahl der Nutzer, Anzahl der neuen Adressdaten, Anzahl der freigerubbelten Felder im Rubbel-Gewinnspiel, Gesamtzahl der gerubbelten Felder bis zu diesem Tag sowie die Gesamtzahl der Teilnehmer bis zu diesem Tag zu sehen und in tabellarischer Form gemäß Design darzustellen.

A.5 Beispiel Screenshots der Programmablaufplan



A.6 Ein Screenshot der Benutzeroberfläche



A.7 Screenshots der Quellcode

```
488 //get states from zip number
489 function getStates(arr) {
490   if (arr !== undefined) {
491     var stateArray = [];
492     var items = [];
493     $.getJSON("plz.json", function (data) {
494       $.each(data, function (key, val) {
495         items.push([val["zipcode"], val["state"]]);
496       });
497       //make an array with states
498       for (let i = 0; i < arr.length; i++) {
499         for (let a = 0; a < items.length; a++) {
500           if (arr[i][4] === items[a][0]) {
501             stateArray[i] = items[a][1];
502           }
503         }
504       }
505       //get how many time states repeat
506       var counts = {};
507       stateArray.forEach(function (x) {
508         counts[x] = (counts[x] || 0) + 1;
509       });
510       //make it sortable array
511       var sortable = [];
512       for (var c in counts) {
513         sortable.push([c, counts[c]]);
514       }
515       //sort descending
516       sortable.sort(function (a, b) {
517         return b[1] - a[1];
518       });
519       //table variables
520       var herkunft = [];
521       var user = [];
522       //table variables
523       var herkunft = [];
524       var user = [];
525       for (let i = 0; i < sortable.length; i++) {
526         const element = sortable[i];
527         herkunft.push(element[0]);
528         user.push(element[1]);
529       }
530       //control group of nonexist states
531       var bundesland = ["Bayern", "Berlin", "Brandenburg", ...];
532       //check which states are not in the game,
533       //and add them with 0 value
534       var a = [];
535       for (var i = 0; i < bundesland.length; i++) {
536         a[bundesland[i]] = true;
537       }
538       for (var i = 0; i < herkunft.length; i++) {
539         if (a[herkunft[i]]) {
540           delete a[herkunft[i]];
541         } else {
542           a[herkunft[i]] = true;
543         }
544       }
545       for (var k in a) {
546         herkunft.push(k);
547         user.push(0);
548       }
549       loadHerkunftTable(herkunft, user);
550       herkunft = []; //reset to null again
551       user = []; //reset to null again
552     });
553   }
554 }
555 }

773 // Dropdown Menu
774 function myFunction() {
775   var menu = document.getElementById("myDropdown");
776   //remove all previously created p's
777   $("#myDropdown").children("p").remove();
778   for (let i = 0; i < databases.length; i++) {
779     console.log(databases[i]);
780     const element = databases[i][1];
781     const _dt = databases[i][0];
782     var p = document.createElement('p');
783     p.className = "button";
784     p.id = element;
785     p.innerHTML = element;
786     p.addEventListener('click', function () {
787       $.ajax({
788         type: 'GET',
789         url: 'connObject.php', data: {
790           db: _dt + '_' + element + '_db5634'
791         },
792         success: function (data) {
793           var dataFromDB = [];
794           dataFromDB = JSON.parse(data);
795           destroyCharts();
796           callAllFunctions(dataFromDB);
797         }
798       });
799     });
800     menu.appendChild(p);
801   }
802   menu.classList.toggle("show");
803 }
804 // Close the dropdown if the user clicks outside of it
805 window.onclick = function (event) {
806   if (!event.target.matches('.dropbtn')) {
807     var dropdowns = document.getElementsByClassName("dropdown");
808     for (i = 0; i < dropdowns.length; i++) {
809       var openDropdown = dropdowns[i];
810       if (openDropdown.classList.contains('show')) {
811         openDropdown.classList.remove('show');
812       }
813     }
814   }
815 }
816 }
817 // Fill Aktivität table
818 function loadTableData(obj) {
819   if (obj !== undefined) {
820     $("#testBody").empty();
821     const table = document.getElementById("testBody");
822     obj.forEach(item => {
823       let row = table.insertRow();
824       let datum = row.insertCell(0);
825       let userProTag = row.insertCell(1);
826       let adressdaten = row.insertCell(2);
827       let anzahlFelder = row.insertCell(3);
828       let gesamtFelder = row.insertCell(4);
829       let gesamtTeilnehmer = row.insertCell(5);
830       datum.innerHTML = dateConversion(item.datum);
831       userProTag.innerHTML = item.userProTag;
832       adressdaten.innerHTML = item.adressdaten;
833       anzahlFelder.innerHTML = item.anzahlFelder;
834       gesamtFelder.innerHTML = item.gesamtFelder;
835       gesamtTeilnehmer.innerHTML = item.gesamtTeilnehmer;
836     });
837   }
838 }
```