JANUARY 8, 2021

SCHERE-STEIN-PAPIER WEB APP ERSTELLEN

EINEN BOT ERSTELLEN, WELCHER GEGEN DEN BENUTZER SPIELEN KANN

VASILY KOZLOV
BICA AG
Buzibachstrasse 45

Contents

Rock-Paper-Scissors-Game

Web-App

Einführung

Ich habe im Rahmen vom Modul 150 ein Project meiner Wahl gestalten können. Ich habe eine PWA entwickelt, mit Hilfe welcher man das klassische «Schere, Stein, Papier»-Spiel gegen einen **BOT** spielen kann. Die Webseite wurde im Internet veröffentlicht und kann unter folgender URL aufgerufen werden: https://vasilyko.github.io/Rock-Paper-Scissors

Technologie-Entscheid

Da ich auf der Arbeit mehrheitlich mit C# arbeite, wollte ich eine Abwechslung zum Arbeits-alltag und habe mich somit für eine Web-App entschieden.

Will ich persönlich ein Mensch bin, dem Glücksspiele grossen Eifer bereiten, habe ich mich für ein Spiel, welches International bekannt ist entschieden: Schere, Stein – Papier. Dank seiner Bekanntheit kann ich meinen User-Traffic möglichst hochhalten. Dies könnte mir in Zukunft eine Monetarisierung bringen.

Anfangs wollte ich meine App auf der Bootstrap-Basis gestalten, doch mein Mitschüler «David Gataric»-welcher lange Erfahrung mit Web-Apps hat, hat mir geraten, das Projekt mit Angular umzusetzen.

Aus dem Grund, dass die Handys heutzutage immer beliebter und zugänglicher werden, habe ich mich entschieden, auf diese Zielgruppe zu konzentrieren. Deswegen werde ich mein Programm mit der PWA-technologie umsetzen.

PWA (Progressiv Web App) eignet sich bestens, um eine Web-Applikation für Smartphone sowie für Desktop zu entwickeln. Unter anderem unterstützt PWA offline-Funktionalitäten.

Ich habe ich mich dazu entschieden, mein Produkt vorwiegend für Handy-Benutzer zu gestalten, aus diesem Grund verfolge ich den Ansatz «**Mobile-First».**

Dies aus dem Grund, dass es um einiges Schwieriger ist, die Information auf einem grösseren Bildschirm darzustellen, als sie übersichtlich auf einer kleineren Fläche darzustellen. (Als Referenz verwende ich das Iphone X - Modell)

Management-Tools

Für mein Project habe ich eine verschiedene Management-Tools verwendet. Diese sowie auch meine Entwicklungs-umgebung, beschreibe ich detaillierter in diesem Kapitel.

GitHub

Den Source-Code pflege ich mit Hilfe von Git-hub. Git-hub ist eine Entwicklungs-Code Management Tool.

Trello

Im Git hub Reporsitory, verlinke ich ebenfalls auf mein Product-Backlog imTrello verlinke. Trello verwende ich aus dem Grund, dass mir dieses Tool in einem der Überbetrieblichen-Kurse vorgestellt worden ist. Da dieses Tool gratis zur Verfügung steht und ich bereits Erfahrungen damit habe, habe ich mich für dieses entschieden.

Entwicklungs-umgebung

Als Entwicklungs-umgebung, habe ich mich für Visualstudio-Code entschieden, da dies die geeignetste Umgebung fürs Arbeiten an Web-Projekten ist. Mir ist die Umgebung bekannt und ich habe bereits mehrere Projekte damit realisieren können. Ebenfalls kann ich verschiedene Formatierungs-Extension herunterladen und es enthält eine Benutzeroberfläche für das Verwalten von Git-Befehlen.
Um den Code-formatter so zu nutzen, wie er bei mir installiert ist, muss man bloss die «Prettier-Code formatter»-Extension. Diese muss lediglich heruntergeladen werden. Danach reagiert diese auf das «Crtl + s»(save)-Event. Danach formatiert es den ganzen Code.
Die IDE Visualstudio-Code kann kostenlos im Web heruntergeladen werden. Danach muss man noch mittels Node.js(welches man ebenfalls im Internet herunterladen muss) folgende zwei Befehle in der Console(im Projekt-Verzeichnis) eingeben:
«npm i»-installiert alle nötigen packages, welche im Projekt genutzt worden sind.(diese Dateien werden im «package.json»-deklariert und mittels diesen Befehl lokal heruntergeladen)
«npm install -g @angular/cli»-dieser Befehl installiert die Angular Umgebung (so können «ng»-Befehle genutzt werden)
Danach kann man in dem Projekt arbeiten.

Vorgehen

Ich habe damit angefangen, dass Default-projekt der Angular CLI anzupassen. Ich habe alles Überflüssige (was für mein Projekt nicht notwendig ist) gelöscht. Ich habe bloss den Header, die Buttons und den footer sowie jeweils die styles übriggelassen, aber für mein GUI angepasst. Anschliessend habe ich meine Bilder, welche ich im Internet ausgesucht habe, als Hintergrund der jeweiligen Buttons gesetzt.

Projektmanagement

Bei der Projektmanagement-Methode habe ich mich für Scrum entschieden. Scrum eignet sich vor allem für Projekte, welche unendlich-langen Entwicklungsprozess haben.

Da mein Produkt später auf einem Server gehostet wird und ich in der Zukunft vorhabe, das Spiel zu erweitern (mehr Elemente nebst Schere, Stein und Papier), eignet sich diese Methode bestens.
Ich habe dabei mich selbst als Product Owner, Scrum Master und Entwickler.

Da wir Wöchentlich unseren Fortschritt der Klasse präsentieren müssen, habe ich die Sprint-Dauer auf «Wöchentlich» gesetzt. Das bedeutet das ich wöchentlich einen Stand haben muss, wo ein Teilstück der Arbeit funktionell vollständig abgeschlossen ist.

Kurzfassung des PA-Berichtes

Im Rahmen der Arbeit im Modul 150 habe mit dem Angular Framework meine Web-Applikation erstellt.

Da der Technologie-entscheid frei zu wählen war und ich eine Web-App (vorwiegend für Mobile-Nutzer) gestallten wollte, habe ich mich für Angular entschieden. Dies aus dem Grund, dass ich viel interessantes über Angular gelesen habe und es passt dank seiner PWA-Bibliothek bestens für mein Projekt. Da ich meine App für Handy-Nutzer optimieren will, (da der meiste Internetverkehr über Smartphones verläuft) liegt der Entscheid für PWA nah.

Ich habe eine PWA für mein Schere-Stein-Papier Spiel initialisiert. Danach habe ich alles, was ich für mein Projekt nicht benötige-gelöscht. Ich habe bloss den Header und die Buttons sowie deren Klassen gelassen.

Als nächstes habe ich die drei Buttons für die drei Elemente erstellt. Ich habe passenden Bilder dazu herausgesucht und in den Hintergrund gesetzt. Ich habe dann die passende Grösse und Reihenfolge für das jeweilige Format (Handy, Tablet oder Desktop) erstellt.

Als ich das passende Format für die jeweilige Auflösung erstellt hatte, habe ich mit dem Designen des Scoreboards angefangen. Ich habe erst ein Rahmen erstellt, in welchem ich danach die Zahlen (Punktestand) darstelle. Damit dem Benutzer klar ist, welche der Punkte seine sind, habe ich die passenden Labels an die jeweilige Seite platziert. Dort wird ebenfalls angezeigt, wer-was gewählt hat.

Als ich das GUI erstellt hatte, habe ich mit der Logik dafür angefangen. Diese habe ich mittels Typescript umgesetzt. Die geschieht folgendermassen: Ich erstelle eine Zufällige Zahl zwischen 1 und 3. Je nachdem welche Zahl generiert wird, nimmt der Computer ein anderes Element (1=Stein, 2=Schere, 3= Papier).

Danach vergleiche ich die Wahl des Benutzers mit der Wahl des BOTs. Je nachdem,

ob der Benutzer gewonnen, Verloren oder unentschieden gespielt hat, kommt ein anderes Gif und Titel. Ich habe ebenfalls keine Kosten und Mühen gescheut und

drei Arrays erstellt, in welchen ich einen passenden Text speichere. Dieser Text wird zusammen mit dem GIF, je nach Resultat ausgegeben.

Des Weiteren habe ich das Resultat-Dialog implementiert. Ich wusste anfangs nicht, wie ich die Ausgabe gestallten sollte. Ich wollte etwas «alert ()»-mässiges implementieren, doch das standart-alert ist einfach hässlich und nicht mehr zeitgemäss. So machte ich mich auf die Suche nach Ideen im Internet, zum Darstellen von Dialogen. So bin ich auf «Sweetalert» gestossen. Ich habe mir danach die Dokumentation vom Sweetalert durchgelesen und versucht in mein Projekt miteinzubeziehen. Passend zum Dialog erscheint ebenfalls ein (lustiges) GIF. Da ich noch neu in Angular bin, ist es mir Anfangs schwergefallen, den Sweetalert sowie all die Komponente richtig mit einzubeziehen. Doch als ich eine Komponente hinzugefügt hatte, kannte ich den Ablauf und konnte somit das Prinzip verstehen und nachvollziehen.

Auf ein Zeil musste ich leider verzichten. Ich habe aus Zeitlichen Gründen nicht mehr geschafft, eine Session für Online-Spiele zu erstellen. Man kann somit (stand 7.01.2020) nur gegen den BOT nicht aber gegen online-Spieler spielen.

Ebenfalls habe ich eine Benutzer-Statistik eingebaut. Der Benutzer sieht unterhalb vom Scoreboard, seine jeweilige gewinn-Chance sowie auch seine Lieblingswahl der jeweiligen Session.

Mockups

Bevor ich mit dem mit dem Code auseinander setzten konnte, habe ich mir Gedanken bezüglich des Aussehens des Programms gemacht.

Beim Erstellen des GUIs habe ich mich an einen Vorsatz gehalten: Weniger ist mehr. Da ich, wie früher bereits erwähnt, den Ansatz «Mobile First» verfolge, habe ich auch das Mockup auf einem Iphone erstellt.

Ich wollte, dass die Bedienung dem Nutzer auch möglichst intuitiv fällt. Aus diesem Grund habe ich mich für drei grosse Buttons auf dem GUI entschieden. Das Scoreboard ist oben-mittig an der Website verankert.

es wird einen Titel geben, welcher in 1-2 Sätzen das Projekt für einen Besucher beschreiben sollte.

Links und rechts vom Score, sind jeweils Labels angebracht, um das Resultat des Benutzers und das des BOTs -zu trennen.

Es sollte auch ein Dialogfeld erstellt werden, welches, nachdem der Benutzer seine Wahl gemacht hat und die Resultate verglichen worden sind, für eine kurze(2-3sek) Zeit erscheinen wird.

Wie es vom meinem Fach-Verantwortlichen (FV) verlangt wurde, werde ich eine Statistik implementieren. Diese wird unterhalb des Scores und oberhalb der Buttons ersichtlich sein.

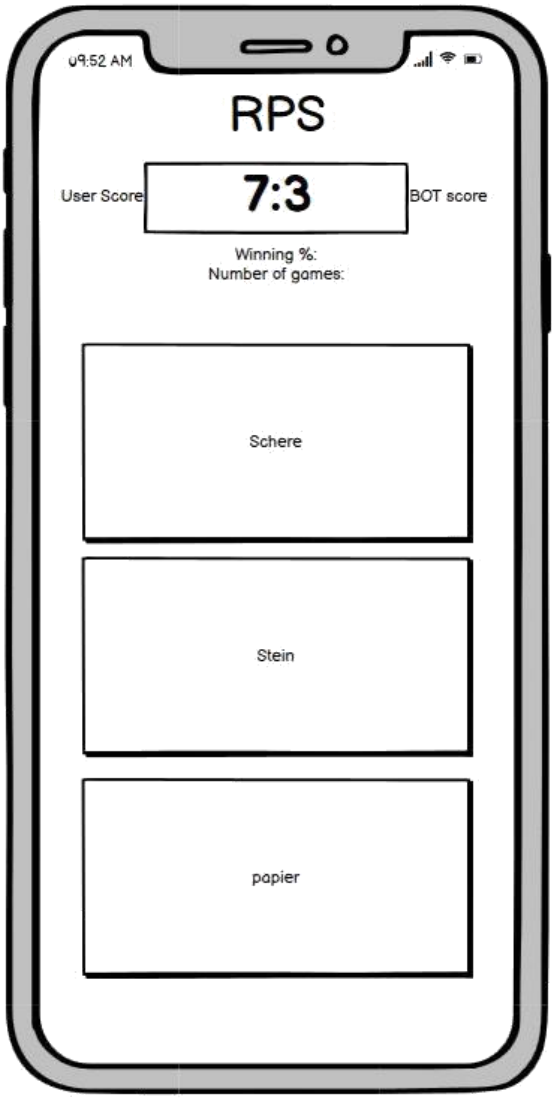


Abbildung 1 Mockup des GUIs

Umsetzung des GUI

Das GUI konnte ich wie geplant umsetzen. Ich habe alles nach dem Mockup gestalten können. Weil mir das GUI gefallen hat (simpel und übersichtlich), habe ich mich entschieden, es für die Desktop-version möglichst ähnlich zu gestallten. Da ich auf dem Desktop-Bildschirm mehr Platz zur Verfügung habe, habe ich die Buttons nebeneinander und grösser erstellt.

Die Resultate-Dialoge erscheinen mit der jeweiligen Meldung (gewonnen, verloren oder unentschieden). Dazu gibt es jeweils ein passendes GIF.

Die Dialog-Texte habe ich in einem Array erfasst und gebe jeweils zufällig einen der Texte aus. Der Dialog verschwindet jeweils nach zwei Sekunden.

Statistik

Eine der Vorgabe durch meinen FV, war es eine Statistik für den Benutzer zu erstellen. Anfangs wollte ich eine DB gestalten, um dort die Daten der Benutzer zu speichern. Da mir jedoch die Zeit schwindet, ich das aber umbedinngt implementieren will, habe ich mich dazu entschieden, die Statistik für die jeweilige Session zu implementieren. Dazu habe ich ein Cookie erstellt. Ich wollte danach das Cookie für eine längere Zeit speichern, ich bin dann aber auf folgendes Problem gestossen: Was, wenn auf dem Computer des Benutzers, seine Kollegen/Verwandte spielen wollen? Ich will nicht das Highscore des jeweiligen PCs-sondern die des jeweiligen Benutzers. Für dies müsste ich dann aber eine DB mit Benutzern erstellen und eine Registration/Log-in erstellen. Dies würde jedoch den Rahmen der Arbeit sprengen. Deswegen habe ich mich für den Moment, für die Session-Statistik entschieden.

Logik-Siegesrate

Für die Gewinnchance habe ich meine Math sowie Menschliche Kenntnisse angewendet. Meine Berechnungsformel sieht daher folgend massen aus:

```
((Gewonnene-Spiele / Gespielte-Spiele) * 100)
```

Anfangs hatte ich die Formel verstauscht, ich habe also die gespielten-Spiele durch die gewonnene-Spiele geteilt. Danach zeigte es mir auf dem UI das unendlich-Zeichen (∞) an. Ich hatte danach die Formel getestet und gemerkt, dass ich die Formel umdrehen muss. Danach hat es mir aber noch immer nicht die gewinn-Chance angezeigt. Nach kurzem Recherchieren im Internet, bin ich auf eine «Prozent»-Pipe gestossen. Somit musste ich im Backend die Gewinnchance folgendermassen setzen:

```
<h2 class="stats"> Deine Siegesrate: {{this.siegesrate | percent }} </h2>
```

Danach wurde die Gewinnchance wie gewünscht angezeigt.

Was verwirrend aussehen könnte, ist das die Gewinnchance, bei z.B. 2:2, eine Gewinnchance von 40% möglich ist. Dies aus dem Grund, dass spiele, welche als unentschieden ausgegangen sind, nicht zu «Gewinn»-gezählt werden. Ich habe es explizit nicht angepasst-aus psychologischen Gründen.

Sollte der Benutzer sehen, dass er eine kleinere Gewinnchance hat, als er erwartet, so wird dem Benutzer unbewusst mitgeteilt, dass der BOT der bessere Spieler ist. Dies regt den Benutzer hoffentlich dazu an, weiter zu spielen, um den BOT zu besiegen.

Logik-Lieblingswaffe

Der Code, um zu bestimmen, was die Lieblingswaffe des Spielers ist, wird viele interessieren.

Im Verlaufe des Projektes, musste ich feststellen, dass der Benutzer mehr als nur eine Lieblingswaffe haben kann. Er kann zwei oder gar alle drei Elemente gleich oft spielen, es wird nur eines Angezeigt. Dies habe ich aus ästhetischen Gründen so belassen. Ich finde es schöner und besser, wenn es nur eine Lieblingswaffe gibt (Wie das der Name des Labels schon sagt (Einzahl)). Ich habe beschlossen den Stein (da es mein Lieblingselement ist), als Favorisiertes Element zu nehmen. Sprich, sollten mehrere Elemente gleich oft gespielt worden sein, so wird trotzdem Stein als Lieblings-element dargestellt.

GUI

Beim Erstellen des GUIs habe ich mich an einen Vorsatz gehalten: Weniger ist mehr.

Ich wollte, dass die Bedienung dem Nutzer auch möglichst intuitiv fällt. Aus diesem Grund habe ich mich für drei grosse Buttons auf dem GUI entschieden.

Das Scoreboard ist oben-mittig an der Website verankert.

Die Resultate-Dialoge erscheinen mit der jeweiligen Meldung (gewonnen, verloren oder unentschieden). Dazu gibt es jeweils ein passendes GIF.

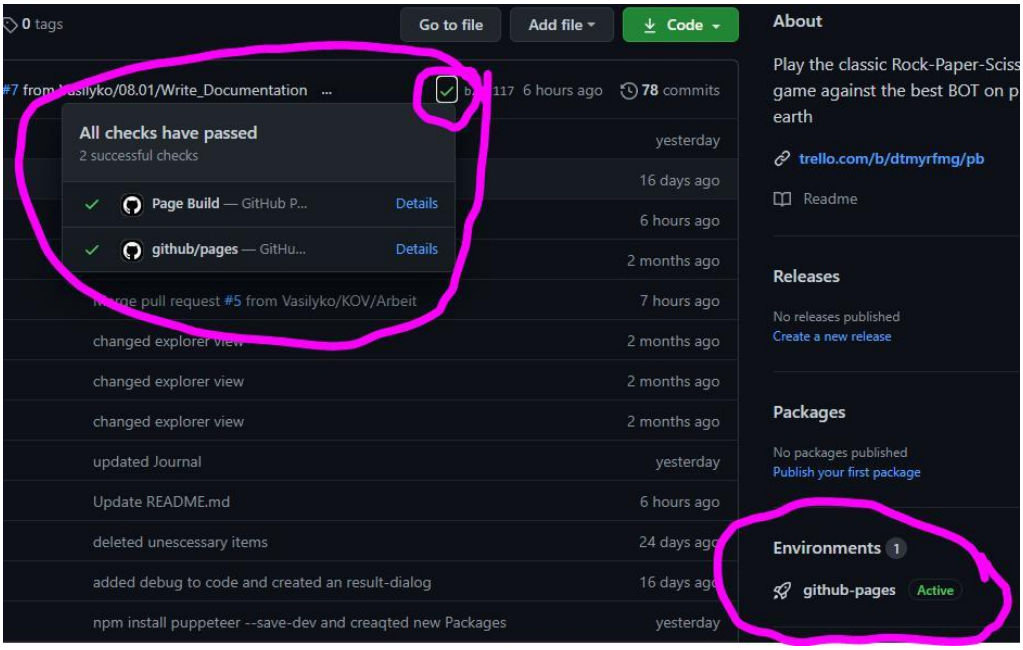
Die Dialoge habe ich in einem Array erfasst und gebe jeweils zufällig einen der Texte aus.
Der Dialog verschwindet jeweils nach fünf Sekunden.

Build

Mein wohl grösstes Problem hatte ich mit CI/CD. Ich hate es in der ersten Phase geschafft, das CI zum Laufen zu bringen (Ich konnte pushen und es hat es geschafft zu builden). Das Ganze habe ich mit Git-hub Actions realisiert. Als ich einst einen weiteren Push tätigte, funktionierte es bereits nicht mehr. Was seltsam ist, denn mein lokaler build konnte erstellt werden.

Eine Vielzahl an commits und stunden-langem Googlen war ich noch immer nicht weiter. Gott sei Dank, bin ich, als meine Hoffnung schon verloren war, auf Git-Pages gestossen. Dies buildet den Code einer Webseite und stellt ebenfalls kostenlos eine Domaine zur Verfügung. ich habe mich in dieses Thema eingelesen und direkt beim ersten Versuch, konnte es meine Web-Seite builden. Dies ist ebenfalls auf meinem Git-Repository ersichtlich. Dies finde ich die passendste build Funktion für mein Projekt.
Danach habe ich mein geschriebenes workflow-deaktiviert.
Solange mein Projekt erfolgreich gebuildet wird, ist es via folgender URL aufrufbar:

https://vasilyko.github.io/Rock-Paper-Scissors
oder lokal mit dem: «ng serve»-Befehl



Reflexion

Es ist mir gelungen, mit einer, für mich sehr mühsamer programmier-Sprache, solch ein großartiges Projekt umzusetzen.

Ich bin mehr als stolz auf mich selber. Ich konnte das ganze Projekt praktisch ohne die Hilfe von 3. Personen umsetzen. Es wurden nicht alle Ziele, welche vom Herrn Imboden vorgegeben worden sind, zu 100& erfüllt werden. Ich musste hier paar Kompromisse eingehen.
Für die Zukunft nehme ich mir 2 Sachen auf den Weg:
1. Ich werde nie wieder für ein Angular-Projekt, eine Git-Action erstellen.

2. Ich werde die Dokumentation versuchen parallel mit dem erarbeiteten Material zu führen. Dies aus dem Grund, dass ich zum Schluss viel Stress hatte. Ich habe zwar das Projekt fertig entwickelt, für meine Dokumentation habe ich aber nur sehr wenig gemacht. Somit bin ich gegen Schluss enorm unter Druck gewesen. Ich hasse solchen Stress.

Mir fehlte ebenfalls die Zeit, um all die verlangten Features umzusetzen. Nichtsdestotrotz habe ich mein bestes gegeben und so gut wie möglich umgesetzt. Ansonsten konnte ich mein Projekt wie geplant abschliessen. Es hat noch zwar einige Probleme, diese werden aber in naher Zukunft behoben.

Es hat mir viel Spass bereitet, an diesem Projekt zu arbeiten. Ich bin dankbar für das Wissen, welches ich mir erarbeiten konnte.

Auch will ich mein Schlusswort nutzen, um meine Bewertung anzusprechen:

Für jemanden, der es von aussen betrachtet, scheint es nach nicht viel zu sein, man muss aber fair sein und berücksichtigen, dass ich mich gewollt für eine Technologie entschieden habe, mit welcher ich keine Erfahrung zum Anfang hatte. Ebenfalls war ich mit einer der einzigen, der diese Arbeit selbstständig im Gegensatz zu den meisten Gruppen erarbeitet habe.

Schlussendlich ist die Bewertung nur subjektiv. Es ist, dass, was der zu Bewertende sieht. Ich für meinen Teil weiss, dass ich viel gelernt habe und viel Blut, Schweiss und das kostbarste-Zeit, investiert hatte.

Glossar

Begriff	Erklärung
PWA	Eine Web Applikation, welche auf jedem Browser kompiliert wird (Mobile wie auch Desktop)
BOT	Umgangssprachlich für Roboter
Mobile-First	Die Applikation erst für Mobile-Geräte entwickeln
Fach-Verantwortlichen (FV)	Mein Betreuer während der ganzen Arbeit
Git-hub	Ein Source-Code Verwaltungs-tool
Product-Backlog	Liste, in der all meine Aufträge, auf einem Kanban Diagramm dargestellt werden.
Default-projekt	Ein Projekt, welches Standartgemäss geladen wird.
CLI	«Command Line Interface», sprich eine Oberfläche, welche nur mit Command-Lines arbeitet.
Angular	Das Web-framework, basierend auf Typescript, welches ich für mein Projekt verwendet hatte.
Refactoring	Den geschriebenen Code überarbeiten(optimieren)
commiten	Den geschriebenen Code lokal abspeichern.
Scoreboard	Engl. Anzeigetafel
Publishen	Im Internet veröffentlichen

Quellenverzeichnis

Alle unten angegeben Links, habe ich am 08.01.2021 aufgerufen und getestet.

Sweetalert: https://sweetalert2.github.io/

Trello: trello.com

Github: github.com
PWA: https://de.wikipedia.org/wiki/Progressive_Web_App

Github-Pages: https://blog.bitsrc.io/deploy-your-angular-project-to-github-pages-7cbacb96f35b

&

https://docs.github.com/en/free-pro-team@latest/github/working-with-github-pages/configuring-a-publishing-source-for-your-github-pages-site#publishing-your-github-pages-site-from-a-docs-folder-on-your-master-branch

Copyright

Rock: https://www.flickr.com/photos/32279598@N02/33507875820

Paper: https://pxhere.com/en/photo/869916

Scissors : https://www.wannapik.com/vectors/3081

Verlierer-GIF: https://media.giphy.com/media/x88yuKqdpWbC0/giphy.gif

Gewinner-GIF: https://media.giphy.com/media/cQNRp4QA8z7B6/giphy.gif

Unentschieden-GIF: https://media.giphy.com/media/3o6ZtgvVil611Pxlmg/giphy.gif

Arbeits-Journal

Datum	Was habe ich gemacht?	Was habe ich gelernt	Wo hatte ich Schwierigkeiten?	Was ist pendent?
12.11.2020	Ich habe das PB erstellt	Ich muss das PB viel einheitlicher Definieren (ganze Funktionen als PBI, nicht nur Teile davon)	Ich habe das PB zu wenig genau definiert. Die Funktionalität vom Produkt ist noch zu mager	Ich muss die PBI anpassen
12.11.2020	Ich habe einen Technologie-Entscheid getroffen (Bootstrap)	Ich hatte durch fehlendes Wissen, nicht von Anfang an die richtige Technologie gewählt.	Ich habe mich nach langem Überlegen doch gegen Bootstrap entschieden, da ich die App Mobile-User freundlich gestalten wollte. Nach einer Diskussion mit Arbeits-Kollegen, habe ich mich für eine Angular-pwa entschieden.	Ich muss die neue Technologie anlegen, da Bootstrap für mich nicht mehr aktuell ist.
12.11.2020	Ich habe ein leeres Git Repo erstellt	Ich habe dies schon mehrere Male gemacht	-	Ich muss das Repo stehts mit Commits pflegen
19.11.2020	Ich habe das Angular Start-Projekt heruntergeladen und PWA Extension hinzugefügt	-	-	
19.11.2020	Ich habe das. yml File erstellet, welches für mich automatisch ein build anstosst	Wie man eine Workflow Datei im GitHub hinzufügt	Die Syntax war für mich bis anhin unbekannt und ich musste mich erst mit dieser auseinandersetzen.	-
19.11.2020	Ich habe einen init Commit ins Repo gepusht	-	-	-

26.11.2020	Ich habe passende Bilder für meine Buttons gesucht	-	-	Ich muss diese Bilder als Hintergrund für die Buttons verwenden
26.11.2020	Ich habe die Dokumentation weiter gepflegt	Ich sollte die Dokumentation immer wieder anpassen. So habe ich weniger Pendenzen.		Immer weiter pflegen.
03.12.2020	Ich habe das Default-Projekt von Angular soweit angepasst, dass ich nur das nötigste für mich übrig gelassen habe		Ich habe anfangs zu viel gelöscht, was die ganze Formatierung beeinflusste	-
03.12.2020	Ich habe die passenden Bilder für das GUI herausgesucht und als Hintergrund bei den Buttons gesetzt.		Ich habe es nicht geschafft, die grösser der Bilder passend darzustellen.	Ich werde bei Gelegenheit mich mit der Platzierung der Bilder befassen.
10.12.2020	Ich habe die Syles der Site angepasst (Schrift, Titel, Schriftgrösse, Schriftfarbe etc.) angepasst		Ich musste erst wieder die HTML Syntax nachschauen, da ich diese wegen mangelhaften nutzen vergessen habe.	
10.12.2020	Ich habe das mat-toggle-slide Modul hinzugefügt.	Wie man ein Modul ins Angular integriert	Ich konnte die Labels beim «toggle» nur jeweils auf eine Seite platzieren (davor oder danach)	
17.12.2020	Ich habe das Score-board für mein Spiel designed	Wie man Daten aus dem Backend im Frontend darstelle.	Mit dem CSS (design): Das darstellen vom Scoreboard, erforderte viel CSS.	Ich muss die Logik hinter dem Scoreboard implementieren.

17.12.2020	Ich habe Spiel-dialoge sowie Resultat-Dialoge mit Sweetalert erstellt.	Wie ich schöne Dialoge im Web darstellen kann (nicht mit «alert ()» Befehl)	Ich habe lange gebraucht, um zu verstehen, wie ich das Modul richtig nutzen kann	-
17.12.2020	Ich habe die Logik für das Scoreboard und somit für die Logik im Spiel implementiert.	Ich kann eine einfache Logik im Typescript umsetzen.	Das Scoreboard hat bei mir anfangs nicht aktualisiert. Es ignorierte teilweise eine Vielzahl an Runden.	-
20.12.2020	Ich habe weiter an der Dokumentation gearbeitet.			
06.01.2020	Ich habe die Statistik (Gewinnchance und Lieblingswahl) implementiert.	Ich kann die percentage-pipe vom Angular anwenden. Ich kann eine Mathematische Formel erstellen.	Anfangs hatte ich als Gewinnchance oft das Ergebnis: «Infinity». Nach langem Debuggen, habe ich festgestellt, dass wenn der Benutzer die erste Runde verliert, es nach meiner Formel nicht aufgeht, da ich sonst durch 0-teile.	Ich muss die Statistik für die Lieblingswahl öfters testen. Es ist noch nicht ausreichend getestet worden.
07.01.2020	Ich hatte versucht, die CI für mein Projekt zu erstellen.	Ich werde nie wieder ein CI/CD erstellen. Eine viel schönere build Funktion, bietet «GitHub-Pages». Mittler dieser können Projekte wie Webseiten über GitHub gepublished werden.	Meine CI/CD konnte nach dem Durchlaufen des Skriptes nicht gebuildet werden.	Ich muss die Dokumentation fertigstellen.
08.01.2021	Dokumentation abgeschlossen	Unter Stress zu arbeiten	-	-