Inhalt der Checkliste

Projektabgabe Join

Bitte erfülle alle Punkte auf dieser Liste, bevor du das Projekt einreichst. Diese **Definition of Done (DoD)** kannst du für alle deine Projekte verwenden.

Merke: Wir werden Join in einzelnen Abschnitten erstellen. Nach jedem "Sprint" erfolgt eine Zwischenabgabe des aktuellen Standes, welcher von den Mentoren abgenommen werden muss. Bitte bei jeder Abgabe den Link zu eurem GitHub Repository beifügen. Bitte achtet darauf, dass euer Repository auf öffentlich gestellt ist.

1. Projekt-Planung

- [ ] Gemeinsames Trello Board einrichten zur Aufgabenverteilung.
  - [ ] Gemeinsame Termine festlegen (Daily Standup).
- [ ] Gemeinsames GIT Repository einrichten.
  - [ ] Projektstruktur aufbauen.
- [ ] Firebase Datenbank einrichten.
  - [ ] Dummy-Daten erstellen. (Grunddatensatz für jeden neuen User)
  - [ ] Realistische Namen / E-Mails etc.
  - [ ] Beachte beim Aufbau der Projektstruktur, dass alle User einschließlich Gast-Login das gleiche Board und die selben Kontakte, Tasks etc. nutzen.
- [ ] Contacts-Seite erstellen
- [ ] Einen Header mit Navbar und Footer erstellen, welche auf jeder Seite eingebunden werden können.
  - [ ] In Contacts Seite einbinden.

2. Verwaltung der Kontakte

User Story 1:

Als Benutzer möchte ich eine übersichtliche Liste von Kontakten sehen und deren Details anzeigen können..

- [ ] Es gibt eine Seite oder einen Bereich für Kontakte
- [ ] Kontakte werden alphabetisch nach ihrem Namen sortiert und ihre E-Mail-Adresse unterhalb ihres Namens angezeigt.

- [ ] Die Liste ist in Abschnitte nach Buchstaben unterteilt, sodass Kontakte, die mit einem bestimmten Buchstaben beginnen, zusammen gruppiert sind.
- [ ] Ein Klick auf einen Kontakt öffnet eine Detailansicht mit dem Namen, der E-Mail-Adresse und der Telefonnummer des Kontakts.

User Story 2

Als Benutzer möchte ich Kontaktinformationen wie E-Mail und Telefonnummer nachschlagen können, um mich mit Kontakten in Verbindung zu setzen.

- [ ] Durch Klicken auf einen Kontakt in der Liste kann ich dessen Detailansicht aufrufen.
- [ ] In der Detailansicht werden alle gespeicherten Informationen wie Name, E-Mail-Adresse und Telefonnummer des Kontakts angezeigt

User Story 3

Als Benutzer möchte ich neue Kontakte zur Liste hinzufügen können, um mit ihnen auf Join zu arbeiten.

- [ ] Es gibt eine "Hinzufügen"-Option oder ein entsprechendes Symbol.
- [ ] Ein Formular öffnet sich, in dem ich Informationen wie Name, E-Mail und Telefonnummer eingeben kann.
- [ ] Form Validation wird nicht durch die Standart-Anzeige dargestellt, sondern durch entsprechende Hinweise in roter Schrift unter dem entsprechenden Inputfeld.
- [ ] Nach dem Ausfüllen des Formulars und Bestätigen werden die Kontaktdaten gespeichert und in der Liste angezeigt.

User Story 4:

Als Benutzer möchte ich Kontakte bearbeiten oder löschen können, um die Kontaktliste aktuell zu halten.

- [ ] In der Detailansicht jedes Kontakts gibt es Optionen zum Bearbeiten und Löschen.
- [ ] Die Bearbeiten-Option öffnet ein Formular mit den bereits gespeicherten Daten des Kontakts, welche angepasst werden können.
- [ ] Die Option 'Löschen' entfernt den Kontakt endgültig aus der Liste.

3. Immer zu berücksichtigen

Allgemein

- [ ] Alle User Stories und Akzeptanzkriterien sind erfüllt.
- [ ] Alle Features funktionieren fehlerfrei und wie erwartet.
- [ ] Vor Abgabe werden mindestens 10 Kontakte mit seriösen Namen hinzugefügt (Dummy-Daten).
- [ ] Alle Funktionalitäten wurden vor Abgabe von den Gruppenmitgliedern manuell getestet mit den aktuellsten Versionen der Hauptbrowser (Chrome, Firefox, Edge, wenn möglich auch Safari).

GitHub-Richtlinien

- [ ] **Pflicht:** GitHub von Anfang an nutzen und pflegen. Denkt dran: Euer GitHub-Profil ist eure Visitenkarte für Arbeitgeber – nutzt diese Chance!
- [ ] Euer **gemeinsames Repository** muss public sein
- [ ] Regelmäßige Commits von jedem Teilnehmer (mindestens ein Commit pro Arbeitssitzung)
- [ ] Verwendet **aussagekräftige** Commit-Messages
- [ ] *.gitignore* verwenden, um unnötige Dateien auszuschließen
- [ ] Nach Abschluss der Gruppenarbeit sollte jedes Gruppenmitglied das Projekt *forken*

User Experience

- [ ] User erhält intuitiv Feedback bei Interaktionen (hover, toast-messages etc.)
- [ ] Alle UI-Elemente (Farben, Abstände, Schatten) entsprechen dem Design-Prototypen in Figma.
- [ ] Transitions auf anklickbaren Elementen liegen zwischen 75ms und 125ms.

Technische Anforderungen

- [ ] Join hat eine MPA-Architektur (Multi-Page-Application mit mehreren separaten Seiten)
- [ ] Dateinamen
  - [ ] beschreibend / aussagekräftig
  - [ ] konsistent
- [ ] Javascript Dateienstruktur / Projektstruktur
  - [ ] Für jede Seite mindestens eine JS-Datei
  - [ ] Eine allgemeine seitenübergreifende JS-Datei
- [ ] CSS- Dateienstruktur
  - [ ] Für jede Seite eine eigene CSS-Datei.
  - [ ] Eine globale CSS-Datei für allgemeingültige Klassen

- [ ] Es gibt keine Fehlermeldungen oder logs in der Konsole
- [ ] Neu erstellter Content wird direkt gerendert

Design

- [ ] Haben Buttons die CSS Eigenschaft cursor: pointer; ?
- [ ] Inputs und Buttons haben keinen Standard-Border (besser border: unset;)

Responsiveness

- [ ] Jede Seite funktioniert bei jeder Auflösung bis min. 320px (Fenster kleiner ziehen)
- [ ] Jede Seite funktioniert sowohl mobile als auch auf Desktop
- [ ] Content-Begrenzung für große Monitore (max-width z.B. bei 1920px / linksbündig)
- [ ] Keine horizontalen Scrollbalken bei kleineren Auflösungen

Formulare

- [ ] Wurde eine Form Validation verwendet?
- [ ] Button deaktivieren während der Ladezeit
- [ ] Form-Validation: Was passiert bei leeren Inputs? (keine Alerts verwenden!)

JavaScript / Clean Code

- [ ] Eine Funktion hat nur eine Aufgabe
- [ ] Eine Funktion ist maximal 14 Zeilen lang (HTML ausgenommen)
- [ ] Deutliche Funktionsnamen
- [ ] Geschrieben in camelCase (Richtig: shoppingCart, falsch; Shopping_Cart) für Dateinamen, Variablen und Funktionen
- [ ] Der erste Buchstabe von Funktionen / Variablen ist **klein geschrieben**
- [ ] 1 oder 2 Leerzeilen Abstand zwischen Funktionen (Konsistent bleiben!)
- [ ] Max 400 LOCs (Lines of Code) pro Datei
- [ ] Dateien sind richtig benannt: index.html, script.js, style.css
- [ ] Ggf. HTML Code in extra Funktion
- [ ] Extra Ordner für templates und Bilder (z.B. img, assets)
- [ ] Statischer HTML Code wird **nicht** über JavaScript generiert
- [ ] Funktionen sind nach JSDoc Standard dokumentiert: https://jsdoc.app/about-getting-started.html

Vermeide diese häufigen Fehler

- [ ] Kein User-Feedback, wenn etwas gespeichert / geändert wird
- [ ] Formvalidation bei Add Contact / Edit Contact fehlt
- [ ] “rauslaufen” von Kontakten und allgemeiner Content

- [ ] verwenden von alerts in der Form Validation