Projektdokumentation

Prüft alle Benutzereingaben client- und serverseitig (Pflichtfelder, Länge, Datentyp). Verwendet sinvolle Datenypen und Speichergrössen zur Speicherung der Informationen.

Ich prüfe alle eingehenden Benutzereingaben serverseitig Mittels middleware. Ich verwende sinvolle Datentypen und nutzte die gleiche Speichergrössen wie in den regex's angegeben.

Verwendet Sessionhandling zur Sitzungsverfolgung und stellt für authorisierte Benutzer neben der Möglichkeit das Passwort zu ändern, zusätzliche Funktionen zur Verfügung.

Ich verwende verwende Sessionhandling mithilfe von JWT. Routen werden auf authorisation überprüft.

Speichert schutzwürdige Informationen sicher und nach neustem Stand der Technik in der Datenbank. Achtet darauf, das der Datenbankbenutzer die für die Kommunikation zwischen Web-Applikation und Datenbank minimal nötigen Berechtigungen besitzt.

Ich speichere das Password als argon2i hash. Argon2 ist momentan der Standard der Hashingalgorythmen. Die Applikation erhält zur Laufzeit eine Datenbank url als env variable, welche nur create, read, update, delete Rechte besitzt.

Verhindert SQL-Injection, Script-Injection und Sessin-Hijacking.

Ich verwende Helmet (library (standard in express)) als middleware gegen SQL-Injections und Script-Injections. Gegen session-Hijacking nutze ich JWT. Ein JWT kann auf seine richtigkeit überprüft werden.

Bietet die Möglichkeit zur Registrierung und Login an der Web-Applikation an. Dazu werden sinnvolle Informationen zum Benutzer erfasst.

Theoretisch ist ein Backend eine Web Applikation, daher ist dies erfüllt.

Sign in: /api/v1/auth/sign_in
Sign up: /api/v1/auth/sign_up

Vom Nutzer werden nur username, password und email entgegengenommen.

Ermöglicht das Erfassen, Ändern und Löschen zusätzlicher Daten über die Webseite in der Datenbank. Dieses Daten können von nicht autorisierten Benutzern angesehen werden.

Nein

Kann einen Testplan für die Applikationen erstellen und überprüft die für dieses Modul relevanten Funktionen der Applikation anhand dieses Testplanes. Kennt Tools zur Validierung von HTML und CSS und validiert seinen / ihren Code.

Habe ich erstellt. Die getesteten funktionen beinhalten: Hashing, Speicherung, Validierung und authentifizierung.
Ich benutze Linter, welche html und css validieren. Im code nutze ich eslint als linter und prettier als formatter. Diese folgen den Richtlinien des AirBnB javascript styleguides.

Setzt für die Projektplanung ein geeignetes Planungstool ein und verwendet dieses aktiv während der Entwicklung des Projektes.

Ich brauche als Plannungstool ZenHub gemischt mit GitHub. Ich erstelle Issues, welche ich dann mit ZenHub in einem Kanban Board herumziehen kann.
Tool wurde durchgängig verwendet.

Kennt Möglichkeiten zur Strukturierung von Quellcode und der Umsetzung unter Berücksichtigung von Codierungsrichtlinien und wendet diese in seinem Projekt an. Kommentiert seinen Code.

Ich liebe Clean Code. Ich habe durchgängig den Styleguide von AirBnB eingehalten.
Ausserdem habe ich mir eine schöne Architektur überlegt und eingesetzt.
Kommentiert ist nur wenig/todos. Grund: In meinem Betrieb lernt man das Clean Code deutlich besser ist als alles zu kommentieren.