Owasp analyse:

Geen sql injection mogelijk, heb gebruik gemaakt van prepare statements. Deze voorkomen dat er sql injection plaats kan vinden.

Afbeelding met tekst, monitor, schermafbeelding, computer

Automatisch gegenereerde beschrijving

Geen encryptie bij het randomwoord, dus eigenlijk kan de gebruiker doormiddel van de opgevraagde api zien wat het te raden woord is.

In het plaatje hieronder is te zien dat het t.getrandomword() een waarde heeft, deze word mee gestuurd. Er is hier niet gebruik gemaakt van encryptie, waardoor het te raden woord indien iemand de api bekijkt vrijgegeven is.

Afbeelding met tekst, schermafbeelding, monitor, computer

Automatisch gegenereerde beschrijving

Broken authentication:

Ik heb hierbij gebruik gemaakt van een simpel id, waardoor je het spel kan hervatten. Ik had geen tijd om te kijken naar een daadwerkelijk authenticatie systeem. Dit zorgt ervoor dat er geen bescherming is.

Daarnaast maak ik geen gebruik van sessie time-outs hiermee zorg ik ervoor dat als je de browser afsluit je direct in het laatste spel terecht komt.

Sensitive data exposure:

Ik heb hiervoor gebruik gemaakt van de secrets voor sonarcube en codecov, deze twee secrets staan op mijn git repository opgeslagen. Hierdoor zorg ik dat deze tokens niet gezien en dus gebruikt kunnen worden door externe party’s.