Slackbot ‘Athena’

08.04.2024 – 06.05.2024

Kandidat
Kent Clark
Betrieb (=Durchführungsort)
Sunrise GmbH
Thurgauerstrasse 101B, PLZ 8152
T 076 555 55 55 (am besten erreichbar)
G +41587775348
M clark.kent@pk19.ch

BerufsbildnerIn/ Lehrfirma
Stammbach Roman
Sunrise GmbH
Thurgauerstrasse 101, 8152 / Glattbrugg
T 0767775348 (am besten erreichbar)
G 0767775348
M roman.stammbach@sunrise.net

Verantwortliche Fachkraft
Stammbach Roman
Sunrise Schweiz
Thurgauerstrasse 61, 8152 / Opfikon
T +41 765717198 (am besten erreichbar)
G 0767775348
M roman.stammbach@pk19.ch

Hauptexperte

T (am besten erreichbar)
G
M

Arbeitsbereiche

-
-
-
-
-
-
-
-
-

Slackbot ‘Athena’

08.04.2024 – 06.05.2024

Ausgangslage

Unser Team setzt Slack als hauptsächliche Kommunikationsschnittstelle ein. Um unsere Erfahrung mit diesem Tool weiter zu verbessern und den Zusammenhalt im Team auch während des Home Office zu stärken, soll im Rahmen der IPA ein Slack-Bot implementiert werden, welcher diverse Aufgaben löst und Informationen schnell und einfach zugreifbar macht.
Unser Team ist sehr verteilt, innerhalb der Schweiz aber auch mit externen Entwicklern weltweit. Die Pandemie hat den Einsatz von Homeoffice noch verstärkt und wir meistern das verteilte Arbeiten hoch effizient. Slack hat sich als Herzstück unserer Kommunikation und auch teilweise als Team-Spirit etabliert. Unser lokales, schweizer Team trifft sich alle zwei Wochen im Büro. Trotzdem lässt diese Art der Zusammenarbeit teilweise Informationslücken. Ein Slackbot, der sich nahtlos in unser tägliches Leben integriert, kann helfen, diese Informationslücken zu reduzieren.

Unser Team entwickelt als Teil von Sunrise die TV-Plattform, die das nahtlose Fernsehen auf Web, Mobile und dem Fernseher ermöglicht. Ausserdem entwickeln wir Apps und die Live-Website für die Plattform. Hier bietet ein hauseigenener Slack-Bot die perfekte Möglichkeit, aktuelle Nutzerzahlen und andere wichtige Daten unserer wichtigsten Projekte auf einen Blick bereitzustellen.

Detaillierte Aufgabenstellung

Im Rahmen der IPA soll ein neuer Slack-Bot entstehen. Die Implementation wird mit TypeScript und NodeJS geschehen. Im Verwenden von Libraries ist die Kandidatin weitgehend frei. Durch das Projekt hindurch wird stets mit Git für die Versionskontrolle gearbeitet. Es wird erwartet, dass die Kandidatin Git versteht und korrekt anwenden kann. Die Kandidatin arbeitet gemäss Git-Flow-Prinzip mit Feature-Branches und Fix-Branches.
Der Rahmen um das Projekt und das Setup von Slack-API-Tokens für die Entwicklung wird bereits vor der IPA erarbeitet.

Die Applikation wird mittels Cloud Build deployed und dann mit Google Cloud Run laufen. Diese Umgebung ist bereits vor der IPA aufzusetzen.

Folgende Funktionen sollen im Slack-Bot enthalten sein:
* Auf Anfrage:
- Aktuelle Users live (Aktuelle Anzahl der Sessions)
- Aktueller Top-Sender
* Wöchentlich:
- Zusammenfassung der aktuellen User Accounts und Zunahme/Abnahme deren im Vergleich zur letzten Woche
- Anzahl Minuten TV geschaut, vergleich zur Vorwoche
- Anzahl Minuten TV geschaut am Wochenende (+ Korrelation zum Wetter, optional)

Slackbot ‘Athena’

08.04.2024 – 06.05.2024

Folgend sind die Akzeptanzkriterien definiert.

Als User des Slack-Bots möchte ich:

- die aktuelle Userzahl der Plattform auf Anfrage (mit einem Command) erhalten
- den aktuellen Top-Sender (Sender mit den meisten Zuschauern) auf Anfrage ermitteln
- wöchentlich eine Zusammenfassung der aktuellen User Accounts auf mit Zunahme/Abnahme im Vergleich zur Vorwoche erhalten
- wöchentlich die Streaming-Gesamtzeit aller User aufsummiert erhalten, mit Vergleich zur Vorwoche
- wöchentlich die Streaming-Gesamtzeit des vergangenen Wochenendes erhalten

Für die wöchentlichen Akzeptanzkriterien soll eine automatische Nachricht jeden Montag gesendet werden. Dies soll in einem eigenen Slack-Channel geschehen.

Die Implementation soll während der IPA verhältnismässig getestet werden. Dafür soll die Library `Vitest` verwendet werden, um Unit-Tests zu erstellen. Es werden verschiedene Funktionen (und/oder Klassenmethoden) getestet, um sicherzustellen, dass diese korrekt funktionieren. Ist zum Beispiel eine Funktion nach aussen sichtbar (durch ESModules exportiert), bietet sich an, dafür einen Unit-Test zu schreiben. Die Kandidatin zeigt in der Dokumentation ausserdem auf, wie gross die Testabdeckung ist. Hierfür bietet Vitest ebenfalls Funktionalitäten, um die Test-Coverage detailliert anzuzeigen. Falls Funktionen nicht getestet werden, kann die Kandidatin erklären, weshalb hier auf Unit-Tests verzichtet wurde.
Integration-Tests werden nicht erwartet.

Es wird erwartet, dass allfällige Fehler, die geschehen könnten (zum Beispiel durch Verbindungsunterbrüche oder im unwahrscheinlichen Fall Nichtverfügbarkeit der API), abgefangen werden, um Crashes zu verhindern. Es wird nicht erwartet, dass jeder mögliche Fehler abgefangen wird, zumal dies mit dem Exception-Modell in Programmiersprachen wie JavaScript unhandlich und unüberschaubar wäre. Es wird lediglich erwartet, dass der Chatbot bei einem Falle eines voraussehbaren Fehlers eine nette Nachricht an den User zurückgibt und nicht einfach abstürzt. Das Exception-Handling soll lokal in der Nähe der werfenden Funktionen geschehen und nicht global als Catch-All-Statement. Allfällige, behandelte Fehler sollen anhand von WinstonJS in ein File geloggt werden.

Die Daten können mit der Google BigQuery-API abgefragt werden. Diese existiert bereits und ist verfügbar. Die Kandidatin hat bereits den API-Key und hat einige erste API-Requests durchgeführt, um sich auf die IPA vorzubereiten. BigQuery ist ein online Data Warehouse, auf welchem verschiedene Daten, wie zum Beispiel aktive Userzahlen, automatisch und durchgehend gespeichert werden. Es kann mittels einer SQL-ähnlichen Syntax auf gespeicherte Daten in BigQuery zugegriffen werden.

Slackbot ‘Athena’

08.04.2024 – 06.05.2024

Die Kandidatin erstellt eine technische Dokumentation in unserem Confluence (in kleinem Umfang), wo das Wichtigste über die Implementation zusammengefasst wird. Es sollte hauptsächlich auf die Architektur und Funktionalitäten eingegangen werden. Die Kandidatin setzt mindestens zwei Diagramme gezielt ein, um die Funktionalitäten und Architektur zu erklären. Es wird erwartet, dass ein Diagramm das umgebende System und die Integration vom Slack-Bot aufzeigt und dass mindestens ein Diagramm in Form eines Flow-Diagramms eine der MVP-Funktionalitäten darstellt. Die Diagramme sollen kurz erläutert und passend in die Dokumentation integriert werden.

Gängige TypeScript-Konventionen werden befolgt. Variablen-, Typ- und Funktionsnamen sind verständlich und aussagekräftig gewählt. Falls sinnvoll, werden Teile vom Code mit Kommentaren erläutert. Für die Code-Styleguide wird eine ESLint-Konfiguration verwendet. Diese wird noch vor der IPA definiert.

Der Code ist sauber strukturiert und wenn möglich modular aufgebaut. Die Architektur wird so gewählt, dass der Code zu einem späteren Zeitpunkt verhältnismässig einfach zu pflegen und zu erweitern ist. Globaler oder weitreichend globaler, mutable State wird vermieden. Abstraktionen wie Typen, Funktionen und Klassen werden mithilfe von ECMAScript-Modulen logisch aufgeteilt.

Mittel und Methoden

Die Kandidatin arbeitet während der IPA mit ihrem Apple MacBook und der IDE ihrer Wahl. Die Entwicklung des Slack-Bots läuft lokal ab, die Kandidatin erstellt nach eigenem Ermessen Code-Commits und pusht diese regelmässig auf das Bitbucket-Repository. Der Slack-Bot wird in der Programmiersprache TypeScript mit NodeJS entwickelt. Die Kandidatin setzt sich bereits vor der IPA mit den Slack-API-Tokens auseinander.

Vorkenntnisse

Die Kandidatin hat bereits einige Erfahrung mit JavaScript und TypeScript. Sie hat sich bereits während der Vorarbeiten mit der Slack-Bot API auseinandergesetzt. Sie kennt Git und wie man sauber damit arbeitet. Sie kennt gängige Diagrammtypen für die Dokumentation und kennt verschiedene Test-Methoden.

Vorarbeiten

Die Kandidatin wird in den Vorarbeiten bereits die Slack-API-Tokens einrichten, ein Projekt erstellen und dieses in Slack integrieren. Sie befasst sich vor der IPA auch mit dem Deployment auf Google Cloudrun, worin wir sie natürlich unterstützen. Sie eignet sich während den Vorarbeiten Wissen in der Verwendung der Slack- und BigQuery-API an.

Slackbot ‘Athena’

08.04.2024 – 06.05.2024

Neue Lerninhalte

Die Kandidatin hat noch keine Erfahrung im automatisierten Testen von JavaScript-Applikationen mit Vitest und wird hier auf Neues treffen. Das Team hat Erfahrung damit und kann die Kandidatin hier während der IPA unterstützen, falls Probleme auftreten würden.

Arbeiten in den letzten 6 Monaten

Die Kandidatin konnte in den letzten 6 Monaten Erfahrungen in der Entwicklung von TypeScript und React sammeln. Ausserdem hat sie Kenntnisse im Testing zusammen mit Gerkhin und ChatGPT gesammelt. Eine Form von AI ist für den Slack-Bot geplant, jedoch nicht Teil der IPA.

Individuelle Kriterien

Auf den folgenden Seiten werden die individuellen Kriterien aufgeführt, welche durch die verantwortliche Fachkraft für diese IPA festgelegt wurden.

Individuelle Kriterien

Leitfrage 1	**Implementation** Der Slack-Bot erfüllt die Anforderungen der Aufgabenstellung: * Auf Anfrage: - Aktuelle Users live (Aktuelle Anzahl der Sessions) - Aktueller Top-Sender * Wöchentlich: - Zusammenfassung der aktuellen User Accounts auf MySports/yallo TV und Zunahme/Abnahme deren im Vergleich zur letzten Woche - Anzahl Minuten TV geschaut, vergleich zur Vorwoche - Anzahl Minuten TV geschaut am Wochenende (+ Korrelation zum Wetter, optional)
Gütestufe 3	Der Slack-Bot erfüllt alle nicht-optionalen Anforderungen.
Gütestufe 2	Der Slack-Bot erfüllt 3-4 von 5 Anforderungen.
Gütestufe 1	Der Slack-Bot erfüllt 1-2 von 5 Anforderungen.
Gütestufe 0	Der Slack-Bot erfüllt keine Anforderungen.

Notizen

Individuelle Kriterien

Leitfrage 2	**Code-Qualität** * Der Code ist sauber, gut strukturiert und verständlich. * Der Code folgt den gängigen Konventionen und ist konsistent. * ESLint und Prettier werden konfiguriert und korrekt angewendet. * Type-, Variablen-, und Funktionsnamen werden sinnvoll und verständlich gewählt. Zum Punkt ESLint und Prettier: Der Code weist zum Zeitpunkt der Abgabe keine grösseren Mängel auf und ist gemäss der Default-Konfiguration von Prettier formatiert.
Gütestufe 3	Alle vier Anforderungen erfüllt.
Gütestufe 2	Drei Anforderungen erfüllt.
Gütestufe 1	Eine bis zwei Anforderungen erfüllt.
Gütestufe 0	Keine Anforderungen erfüllt. Der Code hat offensichtliche, gröbere Mängel und ist nicht konsistent. TypeScript wird nicht konsequent und inkorrekt angewendet. Der Code ist nicht formatiert und ESLint-Regeln werden nicht eingehalten.

Notizen

Individuelle Kriterien

Leitfrage 3	**Architektur** * Der Code ist so aufgebaut, dass er ausserhalb der IPA zu einem späteren Zeitpunkt dementsprechend weiterentwickelt werden kann. * Verschiedene Funktionen, Typen (und Klassen, falls angewendet) werden logisch in ECMAScript-Module aufgeteilt. * Die Kandidatin kann Entscheidungen in der Architektur begründen.
Gütestufe 3	Alle drei Anforderungen erfüllt.
Gütestufe 2	Zwei von drei Anforderungen erfüllt.
Gütestufe 1	Eine von drei Anforderungen erfüllt.
Gütestufe 0	Keine Anforderungen erfüllt.

Notizen

Individuelle Kriterien

Leitfrage 4	**Technische Dokumentation** * Wo aussagekräftiges Naming von Funktionen alleine nicht für das Verständnis ausreicht (selbstdokumentierend), oder Teilprozesse komplex sind und nicht ausgelagert werden können, sind diese mithilfe eines Kommentars erklärt. * Die technische Dokumentation auf Confluence (nach Aufgabenstellung) ist verständlich und nachvollziehbar: - Es werden kurz die umgebenden Systeme erwähnt. - Die Kandidatin kann die Architektur des Slackbots erklären und begründen. - Die Kandidatin arbeitet mit mindestens zwei verschiedenen Diagrammen: - Diagramm zu den umgebenden Systemen und wo der Slackbot darin integriert wird. - Flow-Diagramm zu mindestens einer MVP-Funktionalität beschrieben in I1.
Gütestufe 3	Fünf bis sechs Anforderungen erfüllt.
Gütestufe 2	Drei bis vier Anforderungen erfüllt.
Gütestufe 1	Eine bis zwei Anforderungen erfüllt.
Gütestufe 0	Keine Anforderung erfüllt.

Notizen

Individuelle Kriterien

Individuelle Kriterien

Leitfrage 5	**Anwendung von Git** * Die Kandidatin arbeitet mit Git und versteht die grundlegenden Konzepte. * Es wird gemäss Aufgabenstellung mit Feature- und Fix-Branches gearbeitet. * Branches sind sinnvoll benannt * Die Commit-Messages sind verständlich und ausdrucksvoll formuliert.
Gütestufe 3	Alle vier Anforderungen erfüllt.
Gütestufe 2	Zwei bis drei Anforderungen erfüllt.
Gütestufe 1	Eine Anforderung erfüllt.
Gütestufe 0	Keine Anforderung erfüllt.

Notizen

Individuelle Kriterien

Leitfrage 6	**Fehlerbehandlung** * Geworfene Exceptions werden in der Applikation behandelt. * Dem Benutzer wird im Falle einer nicht behebbaren Exception eine klare Fehlermeldung angezeigt. * Fehler werden anhand von WinstonJS in ein File geloggt. * Die Applikation läuft stabil und stürzt nicht ab. * Es werden keine Catch-All Statements verwendet.
Gütestufe 3	Alle fünf Anforderungen erfüllt.
Gütestufe 2	Drei bis vier Anforderungen erfüllt.
Gütestufe 1	Eine bis zwei Anforderungen erfüllt.
Gütestufe 0	Keine Anforderungen erfüllt.

Notizen

Individuelle Kriterien

Leitfrage 7	**Testing** * Es werden mithilfe von Vitest Unit-Tests erstellt * Wichtige Funktionen werden getestet und dokumentiert * Es wird die Test Coverage dokumentiert * Abdeckungslücken, falls vorhanden, werden identifiziert und ebenfalls dokumentiert.
Gütestufe 3	Alle vier Anforderungen erfüllt.
Gütestufe 2	Drei von vier Anforderungen erfüllt.
Gütestufe 1	Eine bis zwei Anforderungen erfüllt.
Gütestufe 0	Keine Anforderung erfüllt.

Notizen