TECHNISCHE UNIVERSITÄT
CHEMNITZ

Volere Snow Cards

Softwaretechnikpraktikum - Meilenstein 1

Fakultät für Informatik
Professur Softwaretechnik

Eingereicht von: Gruppe 5
Einreichungsdatum: 20.11.2022

Betreuerin: Prof. Dr. Janet Siegmund
Betreuer: Dominik Gorgosch

Zusammenfassung

Die Zeit während des ersten Meilensteins stand in unserer Gruppe ganz im Zeichen des Kennenlernens, der Projektersterstellung und der Aufgaben- und Rollenverteilung. In unserer Gruppe haben wir einen wöchentlichen Termin vereinbart, an dem wir unsere Gruppen-Meetings abhalten. Dafür haben wir uns immer in einem Seminarraum in der Uni getroffen.

Zunächst haben wir uns über die Rollenvergabe und um den Arbeitsablauf in unserer Gruppe Gedanken gemacht. Dazu haben wir uns in unserem ersten Meeting gegenseitig vorgestellt und festgehalten, wer welche (Programmier-)Fähigkeiten und Kenntnisse hat. Louis soll in unserer Gruppe die Rolle des Product Owners übernehmen und in Kontakt mit dem Kunden treten. Im weiteren Verlauf haben wir uns schon grobe Gedanken zu Programmiersprachen und Frameworks, die wir verwenden wollen, gemacht. Nach der Veröffentlichung des ersten Meilensteins haben wir uns gruppenintern nochmal in vier Zweiergruppen eingeteilt, die jeweils die Requirements für die vier Systemteile erarbeitet haben. In einem weiteren Meeting hat dann jede Gruppe die Requirements, mit den ganzen Aspekten für die Snow Cards, vorgetragen. Danach haben wir als gesamte Gruppe entschieden, welche Requirements wir übernehmen und was wir eventuell an Formulierungen etc. für die Snow Cards ändern müssen. Die fertigen, ausformulierten Requirements wurden an Max übergeben, der daraus die Snow Cards mithilfe des Templates generiert hat. Unser Projekt werden wir auf GitHub (`https://github.com/mlinke-ai/kev.in`) hosten, wo wir die Protokolle der Meetings (Sami erstellt während der Meetings immer ein Protokoll) und im späteren Verlauf auch unseren Code hochladen werden. Einen ersten Prototyp unserer Projektstruktur hat Simon bereits erarbeitet und in Github gepusht. Unser Projekt wollen wir mit einer Kombination aus dem Python Framework „Flask“ (`https://flask.palletsprojects.com/en/2.2.x/`), der JavaScript Bibliothek „Svelte“ (`https://svelte.dev/docs`) und einem Datenbankensystem realisieren.

Der erste Meilenstein legt den Grundstein eines erfolgreichen Softwareprojektes und einer guten Gruppenarbeit. Mithilfe der Snow Cards konnten wir erarbeiten, was wir von dem Projekt erwarten und können später immer wieder überprüfen, ob wir uns im Arbeitsprozess in die richtige Richtung bewegen. Es ist auch wichtig gewesen, sich gegenseitig im Team kennenzulernen und zu wissen wer welche Fähigkeiten hat, um die spätere Aufgabenverteilung besser organisieren zu können.

Viele von uns haben bisher nur vereinzelt Programmiererfahrung gesammelt und für alle von uns ist es das erste Mal, ein größeres Projekt nur mit anderen Studenten umzusetzen. Auch dadurch, dass alle unterschiedliche Kenntnisse haben, ist es schwierig, sich zu koordinieren und verschiedene Rollen in der Gruppe festzulegen. Es ist auch schwierig die Rollen dann in der Realität umzusetzen. Aktuell wissen wir noch nicht so richtig, wie und an welcher Stelle wir mit dem Programmierprojekt anfangen sollen. Dadurch, dass alle Leute auch einen unterschiedlichen Stundenplan haben ist es manchmal schwierig einen gemeinsamen Termin für die Gruppenarbeiten zu finden. In unserem Team als Gesamtes gibt es glücklicherweise gute Kenntnisse in verschiedenen Bereichen der Programmierung, sodass wir uns immer gegenseitig helfen und beraten können. Dadurch, dass wir uns alle gut verstehen haben wir auch ein sehr gutes Gruppen-Klima und alle sind motiviert an dem Projekt voran zu kommen.

Login System

Requirement-ID: **1.1** Requirement Type: **functional** Event: **login**

Description: **button press triggers the login process**

Rationale: **pressing the login button starts the login process**

Originator: **user, customer, administrator**

Fit Criterion: **shibboleth system prompts for user credentials**

Customer Satisfaction: **5** Customer Dissatisfaction: **5**

Dependencies: **-** Conflicts: **-**

Materials: **shibboleth documentation**

History: **created: 09.11.2022**

Requirement-ID: **1.2** Requirement Type: **functional** Event: **login**

Description: **user has the option to change his/her password**

Rationale: **when forgetting his/her password the user can request the system to send an e-mail to change the password**

Originator: **user, administrator**

Fit Criterion: **all users can successfully change their passwords**

Customer Satisfaction: **5** Customer Dissatisfaction: **3**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 09.11.2022**

Requirement-ID: **1.3** Requirement Type: **non-functional** Event: **login**

Description: **the front page has an intuitive design**

Rationale: **lowering the hurdles for the user to start using the platform**

Originator: **user**

Fit Criterion: **a test group of users spots the login button in less than 3 seconds on average**

Customer Satisfaction: **2** Customer Dissatisfaction: **3**

Dependencies: **-** Conflicts: **-**

Materials: `https://m3.material.io`

History: **created: 09.11.2022**

Requirement-ID: **1.4** Requirement Type: **non-functional** Event: **login**

Description: **the front page has an appealing design**

Rationale: **making a good first impression and don't annoy the users**

Originator: **user, customer**

Fit Criterion: **a test group of potential users considers creating a user account**

Customer Satisfaction: **2** Customer Dissatisfaction: **3**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 09.11.2022**

Exercise System

Requirement-ID: **2.1** Requirement Type: **functional** Event: **exercise**

Description: **hints for the programming exercises**

Rationale: **useful to avoid the user getting stuck and give additional information**

Originator: **user, customer**

Fit Criterion: **administrator should be able to add hints to an exercise**

Customer Satisfaction: **4** Customer Dissatisfaction: **3**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 10.11.2022**

Requirement-ID: **2.2** Requirement Type: **functional** Event: **exercise**

Description: **search function for exercises with type filter**

Rationale: **user can search for specific tasks to have a more individual learning experience**

Originator: **user, customer**

Fit Criterion: **all exercises can be found and filtered by type**

Customer Satisfaction: **4** Customer Dissatisfaction: **2**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 10.11.2022**

Requirement-ID: **2.3** Requirement Type: **functional** Event: **exercise**

Description: **log various metrics while the user solves the exercises**

Rationale: **statistics can be created to track the learning process**

Originator: **user, customer**

Fit Criterion: **metrics like time and how often an exercise was solved are logged and can be proofed with test**

Customer Satisfaction: **4** Customer Dissatisfaction: **3**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 10.11.2022**

Requirement-ID: **2.4** Requirement Type: **non-functional** Event: **exercise**

Description: **syntax highlighting in all exercises for both Python and Java**

Rationale: **syntax highlighting leads to a more readable and understandable source code, especially for beginners**

Originator: **user, customer**

Fit Criterion: **syntax is correctly highlighted in various scenarios**

Customer Satisfaction: **2** Customer Dissatisfaction: **4**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created 09.11.2022**

Requirement-ID: **2.5** Requirement Type: **non-functional** Event: **exercise**

Description: **prevent the code validation system from running malicious code**

Rationale: **malicious code can lead to system crashes or remote code executions**

Originator: **customer**

Fit Criterion: **malicious code entered into the validation system does not cause the server to crash or undermines system integrity**

Customer Satisfaction: **2** Customer Dissatisfaction: **3**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 14.11.2022**

Administration System

Requirement-ID: **3.1** Requirement Type: **functional** Event: **administration**

Description: **the platform shall conform to data protection requirements**

Rationale: **the administration system shall not allow access to sensitive user information**

Originator: **user, data protection officer**

Fit Criterion: **federal data protection officer approves the design**

Customer Satisfaction: **1** Customer Dissatisfaction: **3**

Dependencies: **-** Conflicts: **3.2**

Materials: `https://gdpr-info.eu/`

History: **created: 05.11.2022**

Requirement-ID: **3.2** Requirement Type: **functional** Event: **administration**

Description: **the administration system shall provide insights into required metrics**

Rationale: **metrics are necessary to create learning schedules**

Originator: **customer**

Fit Criterion: **all the required metrics can be analyzed in the administration system**

Customer Satisfaction: **5** Customer Dissatisfaction: **5**

Dependencies: **-** Conflicts: **3.1**

Materials: **-**

History: **created: 05.11.2022**

Requirement-ID: **3.3** Requirement Type: **functional** Event: **administration**

Description: **the administration system shall provide the ability to create tasks**

Rationale: **creating tasks is necessary to expand the platform**

Originator: **customer, administrator**

Fit Criterion: **after a year of service the platform has more than just the sample exercises**

Customer Satisfaction: **5** Customer Dissatisfaction: **5**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 07.11.2022**

Requirement-ID: **3.4** Requirement Type: Event: **administration non-functional**

Description: **the administration system shall be intuitive to use**

Rationale: **an unintuitive administration system drives the administrators away**

Originator: **administrator**

Fit Criterion: **an administrator shall perform a task in less then 15 minutes after reading the user manual**

Customer Satisfaction: **3** Customer Dissatisfaction: **4**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 07.11.2022**

Requirement-ID: **3.5** Requirement Type: Event: **administration non-functional**

Description: **the administration system shall work performant**

Rationale: **long response times lower the user experience**

Originator: **administrator**

Fit Criterion: **the administration system returns results after a maximum of one second**

Customer Satisfaction: **3** Customer Dissatisfaction: **4**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 07.11.2022**

Evaluation System

Requirement-ID: **4.1** Requirement Type: **functional** Event: **evaluation**

Description: **the user receives an evaluation after submitting a solution**

Rationale: **the user needs to see an evaluation to understand the scoring**

Originator: **customer**

Fit Criterion: **validation with test data**

Customer Satisfaction: **1** Customer Dissatisfaction: **5**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 05.11.2022**

Requirement-ID: **4.2** Requirement Type: **functional** Event: **evaluation**

Description: **the evaluation system saves results to the database**

Rationale: **learning process is evident and comparison the previous solutions is possible**

Originator: **customer**

Fit Criterion: **validation with test data**

Customer Satisfaction: **2** Customer Dissatisfaction: **4**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 05.11.2022**

Requirement-ID: **4.3** Requirement Type: **functional** Event: **evaluation**

Description: **the evaluation system shall provide a sample solution**

Rationale: **user needs to see a correct solution for learning effect**

Originator: **user, customer**

Fit Criterion: **validation with wrong solutions**

Customer Satisfaction: **5** Customer Dissatisfaction: **4**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 05.11.2022**

Requirement-ID: **4.4** Requirement Type: **non-functional** Event: **evaluation**

Description: **the evaluation system handles malicious user input**

Rationale: **the evaluation system is available most of the time**

Originator: **customer, administrator**

Fit Criterion: **the evaluation system crashes only at 1% of cases when presented with specially crafted or randomly generated user input**

Customer Satisfaction: **2** Customer Dissatisfaction: **4**

Dependencies: **-** Conflicts: **-**

Materials: **-**

History: **created: 05.11.2022**

Requirement-ID: **4.5** Requirement Type: **non-functional** Event: **evaluation**

Description: **the evaluation system shall perform tasks with low latency**

Rationale: **high user experience and satisfaction is ensured**

Originator: **user, customer**

Fit Criterion: **in tests the evaluation system returns after at least 1 second**

Customer Satisfaction: **1** Customer Dissatisfaction: **4**

Dependencies: **-** Conflicts: **-**

Materials: `https://www.selenium.dev/`

History: **created: 05.11.2022**