Skapat av Magnus Freidenfelt, MAPP24

Momenten i en Scrum-sprint

1. Sprint planning

Varje sprint börjar med att teamet väljer ut ett antal user stories som de tror att de kommer hinna med under sprinten. Är det första sprinten i ett projekt väljs ofta user stories ut för att kunna producera en MVP (minimum viable product).

Varje user story bryts sedan ner i mindre, mer konkreta uppgifter som också beskriver ungefär *hur* en uppgift ska göras. En uppgift bör ta max en dag att slutföra för en utvecklare. En sprint bör vara mellan 1-4 veckor lång (oftast minst 2 veckor) och en sprint planning bör ta max ca 2 timmar per sprint-vecka.

2. Daily scrum

Varje dag under sprintens gång samlas scrum-teamet för ett kort möte (max en kvart) i början av dagen, ofta stående. Där går alla medlemmar igenom kort vad de arbetat med dagen innan, vad de tänker göra idag och om de har stött på några hinder som de behöver hjälp med. Efter detta möte fortsätter gruppen att jobba med sina uppgifter resten av dagen.

3. Sprint Review

När alla user stories och tasks anses klara (utefter deras ”definition of done”) görs en sprint review. Då sitter hela scrum-teamet tillsammans med beställaren och redogör för vad man producerat under sprinten och vad som har implementerat i produkten. Beställaren får då chansen att reda ut eventuella frågor och ge feedback på arbetet.

4. Sprint Retrospective

Det sista steget innan sprinten är klar är sprint retrospective. Då går scrum-teamet igenom sprinten som varit och diskuterar vad som fungerat bra, vad som fungerat mindre bra och om det är något som behöver förbättras till nästa sprint.

När sprint retrospective är klart är sprinten också klar och nästa sprint kan påbörjas (med en sprint planning).

Rangordning av user stories

Vid skapandet av projektets user stories satt hela teamet tillsammans och bollade idéer. Några i gruppen hade grundläggande kunskaper om diabetes och ungefär hur insulinanvändning går till, några hade också bekanta med diabetes.

User stories gjordes främst ur användarnas perspektiv, men även ur beställarens och utvecklarnas perspektiv. Alla idéer skrevs ned på kort i Miro. Första indelningen skedde genom att alla stories delades upp efter *must, should, could* och *will not* have samt att inom varje indelning sattes det viktigast kortet högst upp i listan. Det första målet var att producera en MVP. Därefter bedömde hela gruppen tillsammans varje kort utefter svårighetsgrad och hur tidskrävande vi trodde det skulle vara att implementera och varje kort fick en story point mellan 1 och 10.

Till nästa gång skulle jag vilja använda något annat verktyg än Miro (antagligen Trello) och använda Fibonacci-skalan för att bedöma korten, eftersom det verkar vara en vanlig metod på arbetsplatser och jag vill öva mer på det. Utöver det skulle jag även vilja spela poäng-poker för att sätta story points. Jag tror att det kan bli mer nyanserat och minskar risken för att utvecklarna hamnar i ett läge där alla håller med varandra ”för att” utan att göra en helt individuell bedömning.

Från user story till app

Den första bearbetningen av en user story med story points görs i en sprint-planering. Där bryts storyn ner i mindre, konkreta uppgifter.

Ett exempel från vårt arbete är storyn ”s*om användare vill jag kunna skapa och logga in på ett konto med email/lösenord och nå kontot från olika enheter”* som delades upp i följande tasks:

- Skapa UI för registrering och inloggning.
- Skapa backend-delen för att kommunicera med Firebase och kunna skapa nya användare.
- Se till att felhantering finns när inloggning/registrering misslyckas.
- Testa att koden fungerar/att det går att skapa nya användare och logga in.

Uppgifterna plockas sen av utvecklarna under daily scrum. När alla tasks är klara och koden har fått en review av någon annan utvecklare anses user storyn vara klar och kan flyttas till done-högen.

I efterhand ser jag att vi hade kunnat varit mer detaljerade med varje task och tydligare med hur varje task kan anses klar. T.ex. hur ska eventuella errors med inloggningen visas för användaren? Ska vi ha någon check på längd eller liknande för lösenord? Vilken kodstruktur ska vi använda - ska backend-koden ligga i ett repository?