Sprint 5 Review

Group 6: Ade Aiho, Heta Hartzell, Mika Laakkonen, Jonne Roponen

User Interface Localization

In sprint 5, our team implemented language support for four different languages. Users are able to dynamically switch between Finnish, English, Japanese and simplified Chinese, which then updates all the UI elements accordingly. All of the translations are stored in the resource bundle “messages”, which contains translations for every text element in the code that was previously hard coded. All elements, such as labels, buttons, alerts and views, are translated once a language is selected. Also, all element and text placement was refined to fit each translation appropriately. This was necessary since character length and size vastly differed between languages.

A page for choosing a language was created, which appears after the initial welcome page. Users choose a language from a set of four options: Finnish, English, Japanese and simplified Chinese. This page lets you choose only one language, which then translates every UI element to it. Thorough testing was conducted for each language to ensure the proper functionality and placement of each language setting, especially with non-latin languages such as Japanese and simplified Chinese. Some element sizes were changed to fit the translation better and accommodate the chosen language.

All test cases regarding UI were also run and updated to accommodate the new language feature. Testing verified that the application still runs as intended after the implementation of runtime translation.

Database Localization

Our database localization included creating a new LocalizedUser class to accommodate for multi-language support. Users are now able to enter and save their name and role in the four supported languages. This addition required changing the database structure: a new table was created to store localized user data. This table has a composite primary key, which consists of an auto-generated ID and a language field. The non-localized User classes’ ID is used to join the two entities in a many to one relationship: a User can have multiple LocalizedUsers.

Several new methods were added to the User class to fetch a LocalizedUser based on language, fetch the first available LocalizedUser, and to add and remove a LocalizedUser. These methods were necessary to ensure the UI functions and can fetch the correct name translations from the database.

Product Backlog Update

Four new user stories were added to the product backlog, which addressed the new localization features. Users should be able to use the application in their preferred language and it should also handle the input and output data correctly in each supported language. Users should also be able to switch their language dynamically during application runtime, which means a need for language page implementation. The application should also adhere to the users' culture in a way that the user experience stays intuitive and fun.

The amount of planned story points for this sprint were 23 in total, of which all were achieved. After the completion of all the planned user stories shown in image 1 the amount of story points collected is 23.

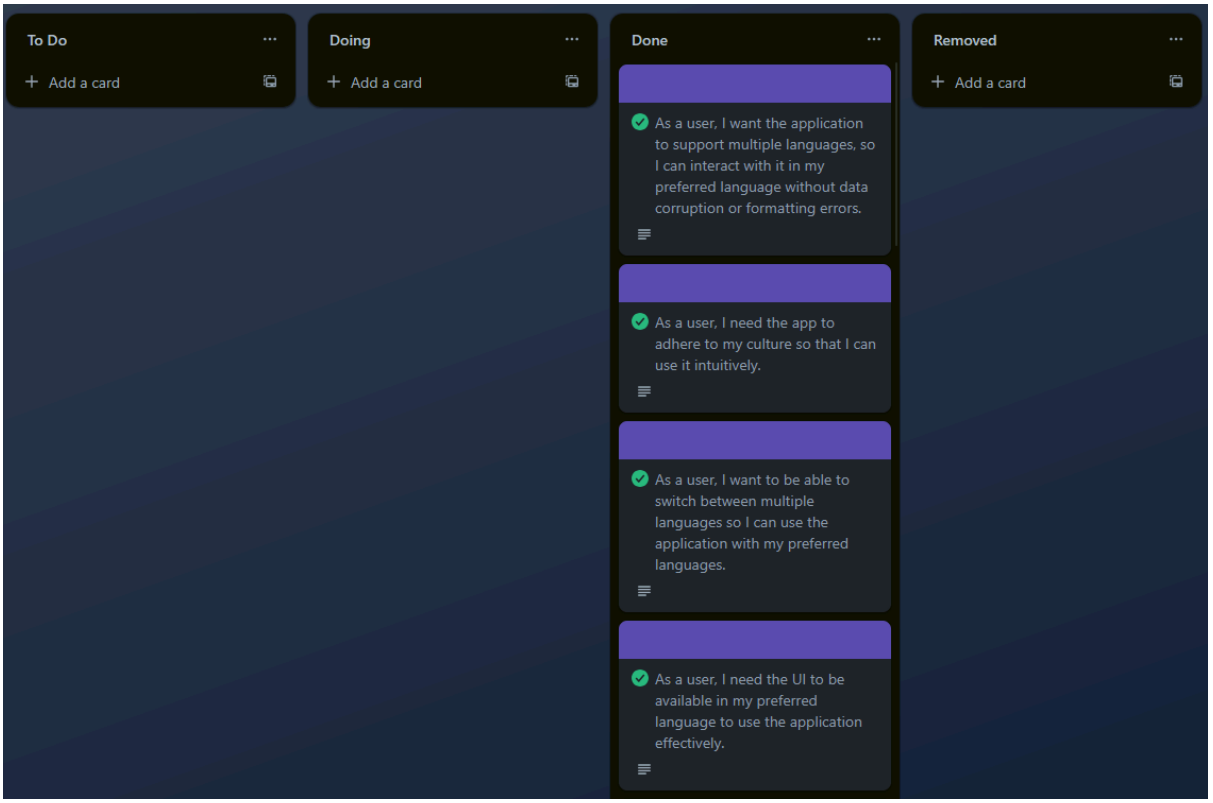


Image 1. sprint 5 product backlog.

The planned user stories to be completed in this sprint were split into several smaller tasks that account for all of the acceptance criteria of each user story.

Sprint 5 backlog status, all the user stories that were split into smaller tasks shown in image 2, tasks 20 in total, which 19/20 were completed

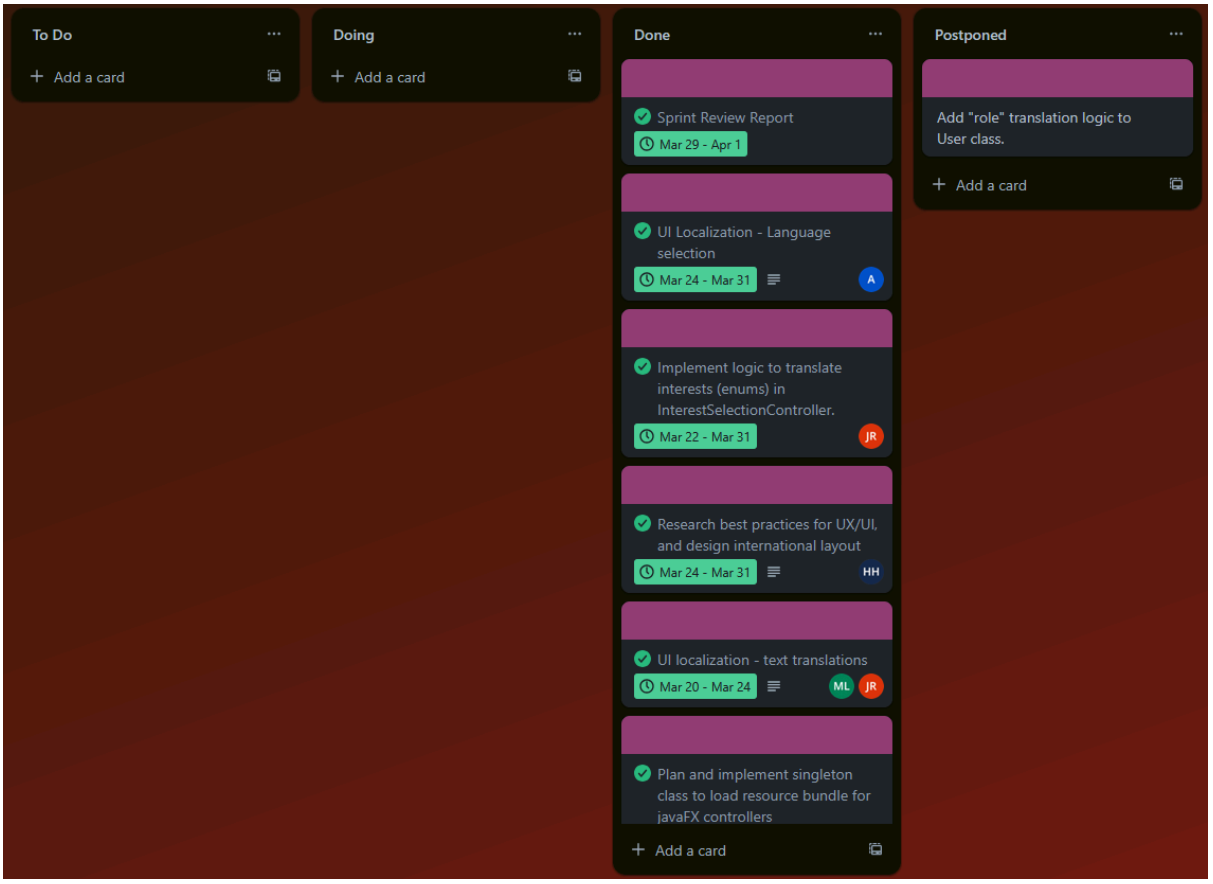


Image 2. sprint 5 backlog.

Localization Resources Identification

All the FXML files and controllers were thoroughly scanned for static UI text elements and compiled in a language bundle. After an initial language file was set up from the original application language Finnish, the subsequent translations were generated based on the Finnish localization.

The resource bundles for the four chosen language options are Messages_en_US.properties, Messages_fi_FI.properties, Messages_ja_JP.properties, and Messages_zh_CN.properties. A unified format for storing the strings in the resource bundles was used for easy and quick navigation for said translation (such as “pair_history=parihistoria”, “pair_history=pair history”, “pair_history=ペア履歴”, “pair_history=配对历史”).

Translations included all elements from UI views, error messages and enum values. In the case of UI views for example this meant that login buttons in the options page were translated, so users can use the application in their own language from the four available choices. For the non-latin languages the elements were studied to fit their locale and cultural needs, for seamless user experience. Warnings were also gathered in the bundles and made to be clear and precise in every possible language option and handled in the controllers of their respective pages. Enums

during the interest selection pages are also gathered in the bundle and translated so they are available in the chosen languages. Enums are stored as numbers in the database but are dynamically translated in the controller during application runtime so all the interest options are correctly mapped to the current language.

Resource bundles for all the languages are easy to navigate and manage in regards to updating older elements or in the case of adding new translations.

Sprint Planning and Review

The sprint goal was defined for multilingual support and the localization plan was made early in the sprint to avoid any unnecessary rework. The whole team participated in both sprint ceremonies, the sprint planning and review. The group went through the sprint 5 requirements and divided responsibilities among all members. The requirements were tailored to align with the goals of our project.

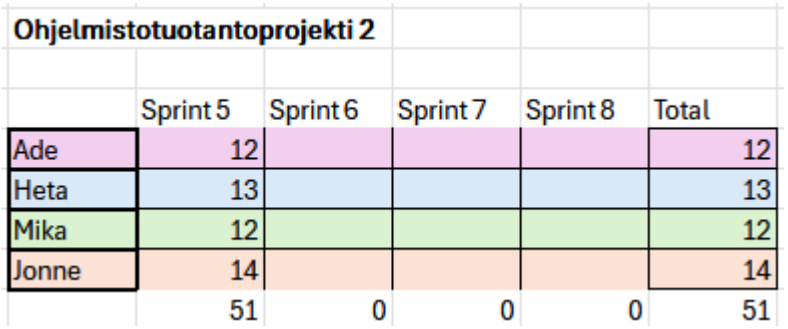
All UI elements now support localization. The application correctly displays and stores unicode input from the user. Non-latin languages Japanese and Chinese were tested and made to fit the layout.

The following table 1 shows the contributions of each team member:

Name	Tasks	Time
Ade Aiho	Language singleton, Functionality integration	12 hrs
Heta Hartzell	Language selection page, localization resources alignment	13 hrs
Mika Laakkonen	Database localization, UI translations, UI-DB integration	12 hrs
Jonne Roponen	Overall UI translations, Resource bundle management, updated controllers and UI tests	14 hrs

Table 1. Individual contributions.

Image 3 shows the excel sheet which displays each team member's contribution.



Ohjelmistotuotantoprojekti 2

	Sprint 5	Sprint 6	Sprint 7	Sprint 8	Total
Ade	12				12
Heta	13				13
Mika	12				12
Jonne	14				14
	51	0	0	0	51

Image 3. image of excel sheet team contributions.