Report

CSM14103 Software Architecture Lab

Fisukortti

Assignment type A

Susanna Rajamäki

Mikko Vallaskivi

16.3.2021

MATEMAATTIS-LUONNONTIETEELLINEN TIEDEKUNTA

HELSINGIN YLIOPISTO

Contents

1 Introduction (Mikko and Susanna)

Fisukortti is an online service, which gathers fishing spots from Finland. Users can browse rivers, lakes, archipelago and shorelines to find fishing spots that suit their need. The spot listing shows important info about the spot, regulations, recommended fishing methods and other nearby spots. Users can search spots per fishing method, fish species, spot type or location. They can also save their searches. Users can review a spot and add pictures to their reviews. Users can add a specific spot to their 'Favorite Spots'-list. Spots can be viewed as a list or listed on a map. Users can login, manage their reviews, and see their favorite spots. Users can share their favorite spots on Facebook, Twitter, Instagram and WhatsApp. The service is free to use and revenue model is based on affiliate sales and ads from an advertising aggregator. The service can be used online, on tablet and on mobile devices.

Extra features are listing the boats available for rental at or near a spot from the Finnish boat rental service Skipperi, listing fishing guides in the area, and users can buy the recommendations of a fishing guide. Users will be able to buy fishing permits for private spots within the service. Users can view custom-made lake and archipelago depth charts if these are available through Navionics API. Users can get access recommendation on the map from Google transport API.

Fisukortti service is meant to be used on free time and it is not critical if it crashes or it has high latencies. Fisukortti is an online service so there is high probability

of security threats. On the other hand, impacts of failure are very low so that is why Fisukortti is overall a low-risk application. The service will be deployed in the cloud where the cloud providers security measures come into effect, reducing the risk of Denial-of-Service attacks and provide a more robust user authentication via cloud providers integrations to OAuth providers. Main risks of this project are on the design of service so that the most important quality attributes are fulfilled. Fisukortti will be designed according to microservice architecture pattern so that many people can work on it and it will be highly scalable and maintainable. On the other hand, when using the microservices architecture additional complexity of creating a distributed system must be deal with. Distributed architecture will slow down development, but this will be tackled by using a planned design.

2 General

Fisukortti has GitHub repository.

2.1 Requirements (Mikko and Susanna)

Fisukortti is an online service, which gathers fishing spots from Finland. The current plan is to focus only on Finland.

Users

Users are individuals in Finland, who are interested in fishing. There are about 1,6 million fishing enthusiasts in Finland.

Requirements

- The service can be used online, on tablet and on mobile devices.
- Users can login.
- Users can browse rivers, lakes, archipelago and shorelines to find fishing spots that suit their need.
- Users can search spots per fishing method, fish species, spot type or location.
- Users can review a spot and add pictures to their reviews.
- Users can add a specific spot to their 'Favorite Spots'-list.
- Users can manage their reviews and see their favorite spots.

- Users can share their favorite spots on Facebook, Twitter, Instagram, WhatsApp and other social media platforms.
- Spots can be viewed as a list or listed on a map.
- The spot listing shows important info about the spot, regulations, recommended fishing methods and other nearby spots.
- The service is free to use to the users.
- Revenue model is based on affiliate sales and ads from an advertising aggregator.

Extra features

- Listing the boats available for rental at or near a spot from the finnish boat rental service Skipperi.
- List fishing guides in the area.
- Users can buy the recommendations of a fishing guide.
- Users will be able to buy fishing permits for private spots within the service.
- Users can view custom-made lake and archipelago depth charts if these are available through Navionics API.
- Users can get access recommendation on the map from Google transport API.

2.2 Quality attributes (Susanna)

An important part of architecture work is to learn to recognize and analyze the quality requirements that architecture design can and should deal with.

2.2.1 Recognized quality attributes

Dynamic or operational quality attributes:

- Scalability is the ability for the system to perform and operate as the number of users or requests increases. Fisukortti must be up and running when there are many users on a sunny Sunday afternoon.

Static quality attributes

- Extensibility is important, because adding new pieces of functionality must be easy.
- Localization so that there is support for multiple languages.
- Maintainability so that it is easy to apply changes and enhance the system.

Cross-cutting quality attributes

- Accessibility so that all users can access Fisukortti, including those with disabilities like visually impaired.
- Authentication because Fisukortti-users can login.
- Usability so that Fisukortti is easy to use.

Open issues

- Quality attributes about authorization, legal, privacy and security issues are still open.

2.2.2 Analyzing of quality attributes

Scalability is handled by using microservice architecture and by using proper cloud services. Scalability is an important requirement when deployment strategy is designed. Choosing a cloud service provider and service level agreement will be critical.

Quality attributes accessibility and usability will affect user interface design more than backend design. Localization is implemented when coding starts. Maintainability can be increased by using clean code principles. These issues are not addressed in this report.

The most important quality attribute at this phase of design work is extensibility. This issue is taken into account in design process.

3 Domain model

3.1 Graphical model (Mikko and Susanna)

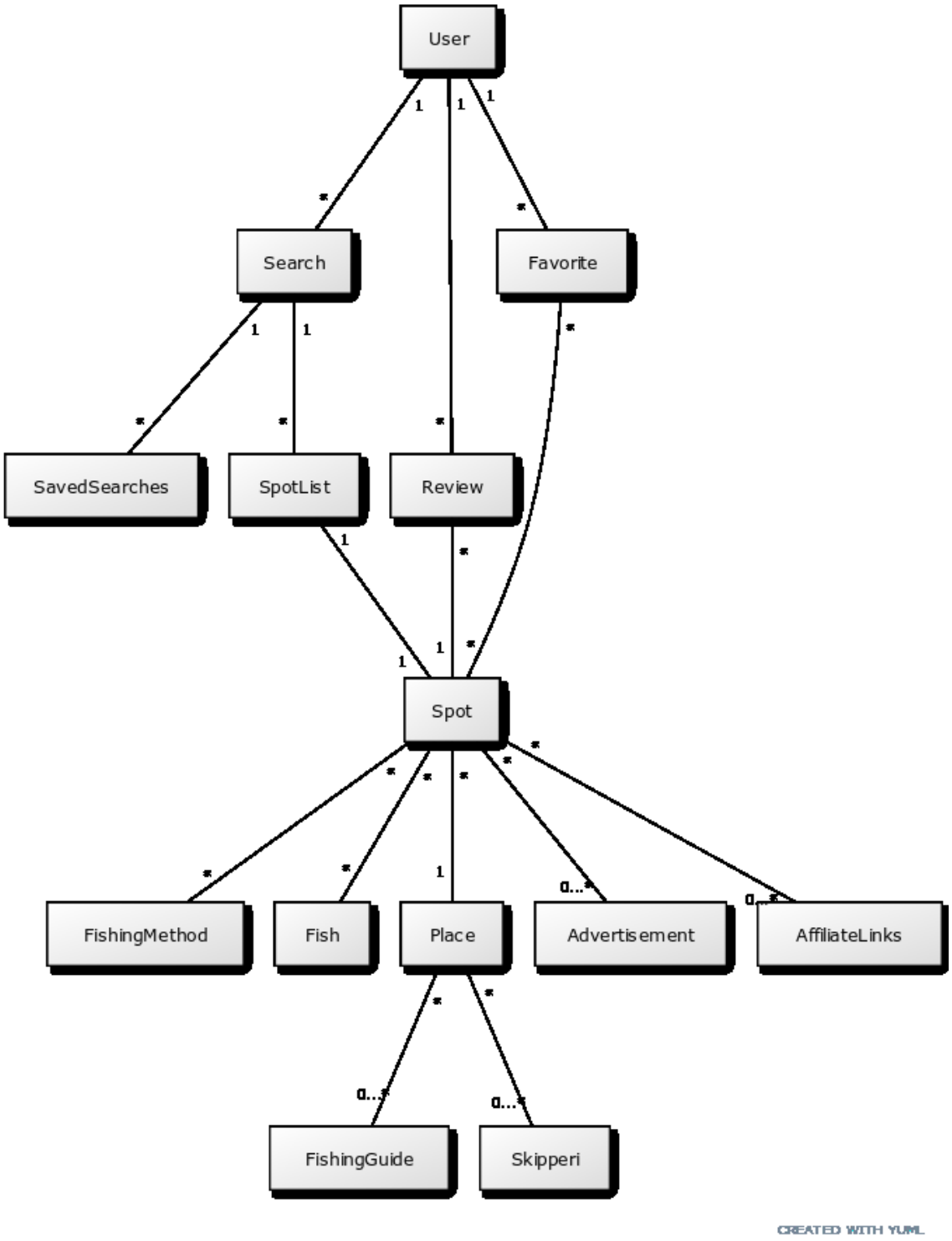

Figure 3-1: UML-diagram

Invariants:

- User has to be unique
- User is required to perform a search with at least 2 search terms out of 3
- Spot has to be unique
- FishingGuide, Skipperi, Advertisement and AffiliateLinks are not mandatory in the Spot or Place.

3.2 Information model (Mikko and Susanna)

- Spot
  - Place to fish, for example a lake, a river, a shoreline, a pond, a creek. Each Spot has at least one Fish, FishingMethod and Place. Spots can have Advertisement and AffiliateLinks.
- SpotList
  - SpotList is the Spot listings displayed based on User's Search terms.
- User
  - The person using the service and searching the Spots to find out where to go fishing.
- SavedSearches
  - Users can save Searches they have done for later use.
- Review
  - User can make Reviews for Spots. Other Users see Reviews when they view the Spot.
- Favorite

- Favorite is User's favorite fishing Spot. User can have many Spots as a Favorite.

- Search
  - Search is used to find Spots. Search can be done by using Place, Fish or FishingType as a search term.
- Place
  - Place is a geolocation to be used in the Search as a search term, for example a city, a county or an area. Places can have FishingGuide and Skipperi.
- Fish
  - Fish is a species that is used in the Search as a search term, for example a trout or a perch. The species list is based on the information provided by Laji.fi.
- FishingMethod
  - FishingMethod is a way to catch fish and it is used in the Search as a search term, for example fly-fishing, trolling, casting and pole fishing.
- FishingGuide
  - FishingGuide is a person offering guiding services at specific Places. The FishingGuides are displayed at the Spot page. There can be multiple FishingGuides available at the specific Place. User can click the FishingGuide info box and will be redirected to the FishingGuide's personal web page.
- Skipperi
  - Skipperi is a Finnish boat rental service. If Place has any available boats, they are displayed at the Spot page.
- Advertisement

- Advertisement is displayed at specific location at the Spot page. Advertisements are from Google AdWords.

- AffiliateLinks
  - AffiliateLinks for lures, gear and equipment are displayed at the Spot page.

3.3 Functionality scenarios

3.3.1 Fly-fishing scenario (Mikko)

Jack is going for business trip to Jyväskylä in first week of June. He knows that the meetings will end at 14:00 so he will have the afternoon and evening to do what he wants. Jack wants to go fly-fishing for brown trout in one of the river rapids near Jyväskylä. Jack searches the listings with keywords 'fly-fishing' and 'Jyväskylä'. He clicks open the first listing for Kuusaa rapids. The listing says that Kuusaa is private, and therefor Jack needs to buy the private permit. The permits are for 24 hours and 12 hours. Kuusaa description states that the flow is still high at beginning of June so a heavier fly rod is recommended which can handle heavy sinking lines. Jack does not have such setup with him. However, in the listing he can see the services of a local guide who rents fishing gear. Jack contacts the guide to book the needed gear, books a permit through phone since the place owner does not have online booking available. Jack then checks the best way to get to Kuusaa, and the fastest and cheapest method of transportation is a taxi. Week later Jack is asked through his own profile to provide a review for Kuusaa and for Juha's Guide services.

3.3.2 Worm scenario (Susanna)

Susan, Emil and Jonathan digged some worms from their yard. They decided to go pole fishing in Espoo. Susan used Fisukortti service with her phone to see where to go. It's everyman's right to go pole fishing without a reel using worm so Fisukortti showed all the water systems in Espoo. Susan added to search terms that they want to use bus to travel and shared their current location. Fisukortti gives water system options that are near a bus stop and timetables how to get there.

3.3.3 Skipperi scenario (Mikko)

Paul looks for a place to fish for perch and zander near his home in Espoo. He searches the spots with search terms 'perch' and 'Espoo'. He is proposed several lakes and shoreline spots in Espoo. Paul runs through the listing and reads the reviews, only to realize that perch fishing from shore is not very effective at this time of the year. However, the perch are scattered among deeper weed beds in around 4 to 6 meters of water. Paul looks at the listing for the spot at Kivenlahti Marina and sees the advertisement to rent a boat through Skipperi for the day. Paul clicks the boat picture and is taken to Skipperi to finalize his rental for the given time. After the rental is confirmed, Paul is returned to the listing for Kivenlahti Marina on Fisukortti and he reads through the reviews, checks the weather forecast for the spot provided by Foreca API, and decides to buy couple of jigs recommended by reviewers through the link to Happy Angler. From the regulations section Paul realizes he needs the national fishing permit, and clicks the

link to the Eräluvat website to buy one. Week later, Paul is emailed by Fisukortti to provide a review of the spot.

3.3.4 Search scenario (Mikko)

Name: Sanna searches for a whitefish spot in Helsinki region.

Initial State: Sanna wants to go fishing for whitefish and looks for spots.

Actors: Sanna, web service

Steps:

1. Sanna opens a web service in web browser.
2. Sanna searches the web service.
3. Web service displays a list matching the search terms Sanna input.
4. Sanna clicks open a listing.
5. Web service renders a place in the browser.
6. Sanna looks at the page.

4 Design model (Mikko and Susanna)

4.1 List of use-cases

Non-registered user can:

- Make a search for spots
  - View list of search results
- View spots
  - Go to Skipperi service from the spot page
  - Go to fishing guide service from the spot page
  - View ads from spot page
- Share spots
- View reviews of the spots
- Create Fisukortti account

Registered user can also:

- Login to Fisukortti account
- Add spot to favorites
- View favorites
- Save searches
- Review spots

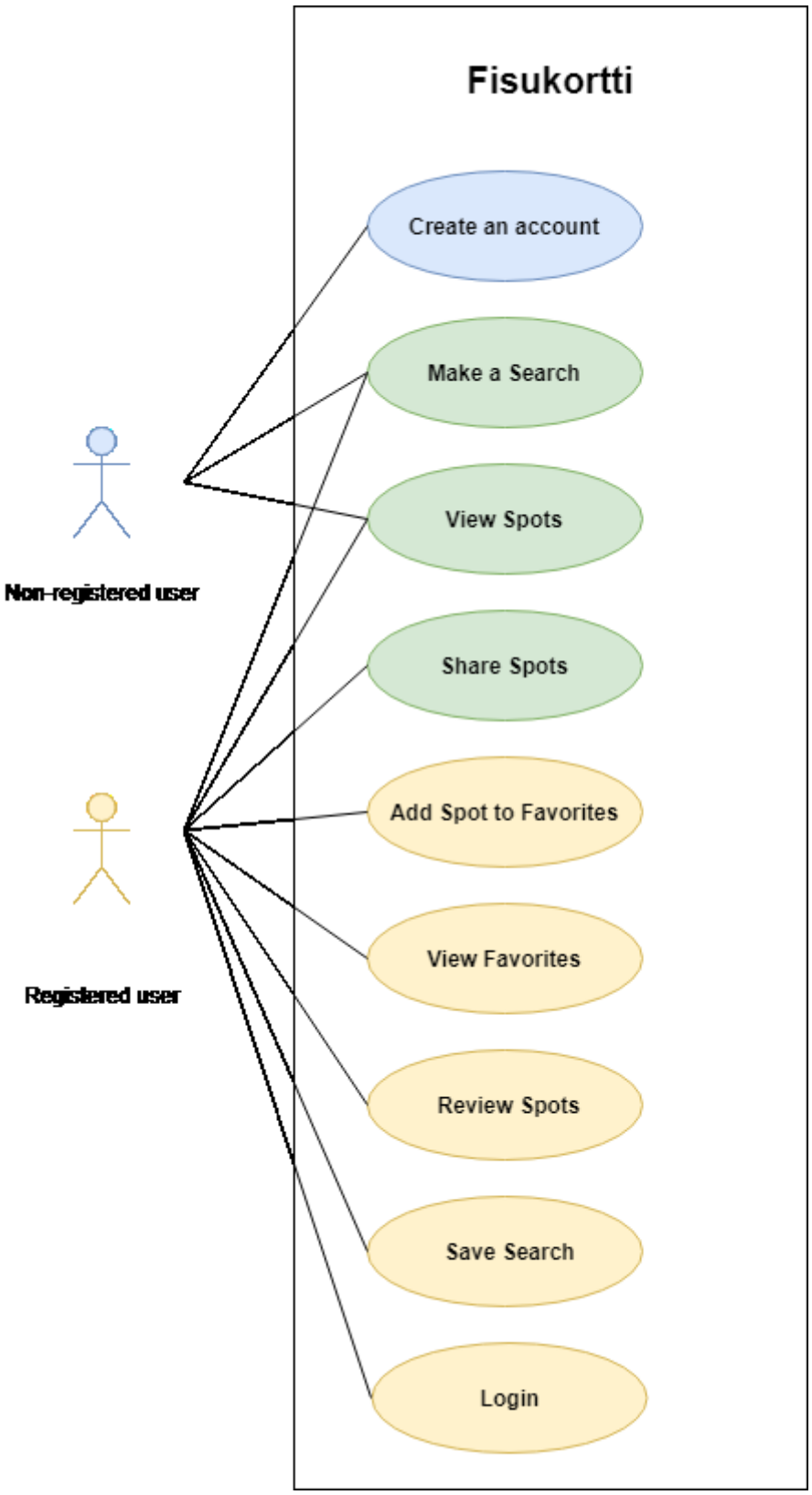

Figure 4-1 Use case diagram

4.2 Functionality scenarios

4.2.1 Functionality scenario: Search

Name: Peter searches pike fishing spots in Nauvo.

Initial State: Fisukortti web service is active. It is July summer holiday season and Peter wants to go fishing. Peter is a unregistered user of Fisukortti.

Actors: Peter

Steps:

1. Peter opens Fisukortti web service in web browser.
2. Peter makes search with terms:
    a. Fish: 'trout'
    b. FishingMethod: 'pike fishing'
    c. Place: 'Nauvo'
3. Peter views list of search results.
4. Peter clicks open and views the spot for 'Granvik'.

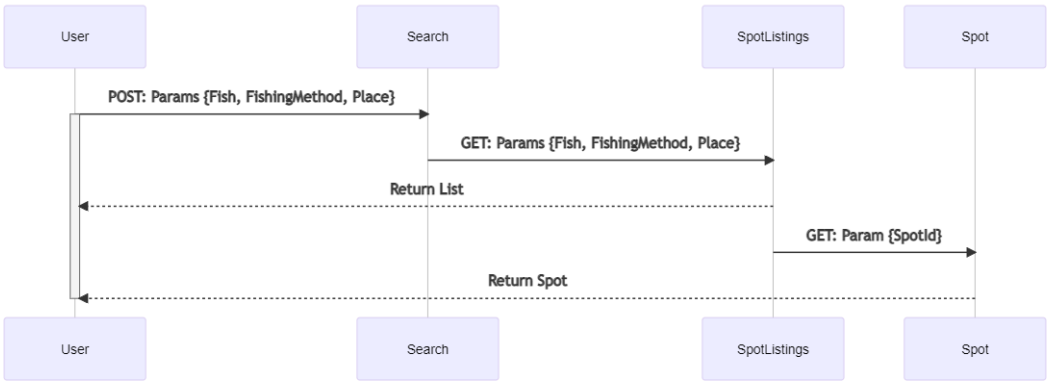

Figure 4-2: Message sequence chart of Search

4.2.2 Functionality scenario: Saved search

Name: Peter saves a search

Initial State: Fisukortti web service is active. Peter is now a registered user of Fisukortti and he wants to save search parameters after search.

Actors: Peter

Steps:

1. Peter saves search.
2. Next time Peter wants to make a same search, he does not have to write parameters.
3. Peter views list of search results.
4. Peter clicks open and views the spot for 'Granvik'.

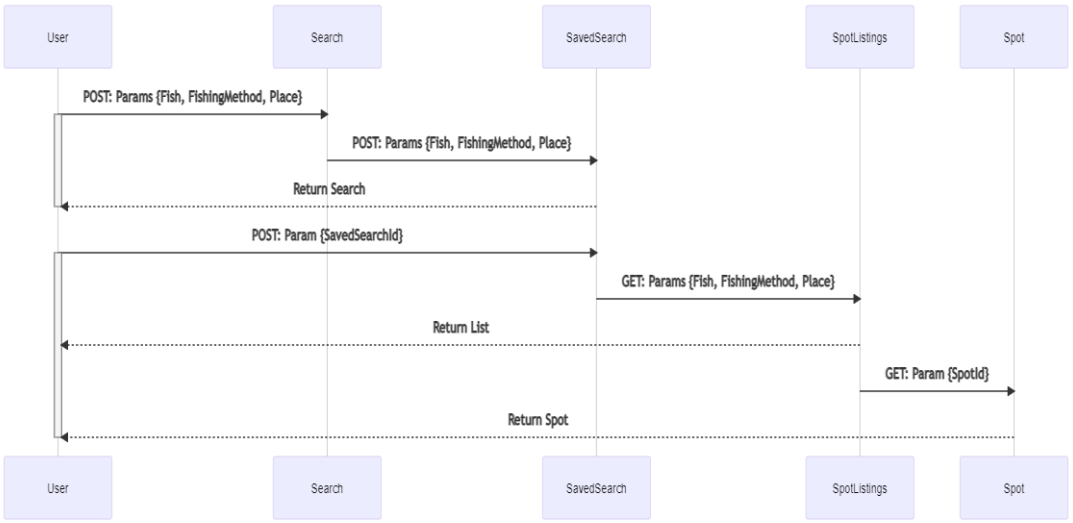

Figure 4-3: Message sequence chart of Saved search

4.2.3 Functionality scenario: Favorites

Name: Peter views favorite spots.

Initial State: Fisukortti web service is active. Peter is a registered user of Fisukortti and he has found a new favorite spot.

Actors: Peter

Steps:

1. Peter adds a spot to favorites.
2. Next time Peter views his favorites.
3. Peter clicks open the spot for 'Granvik' from the favorite list.

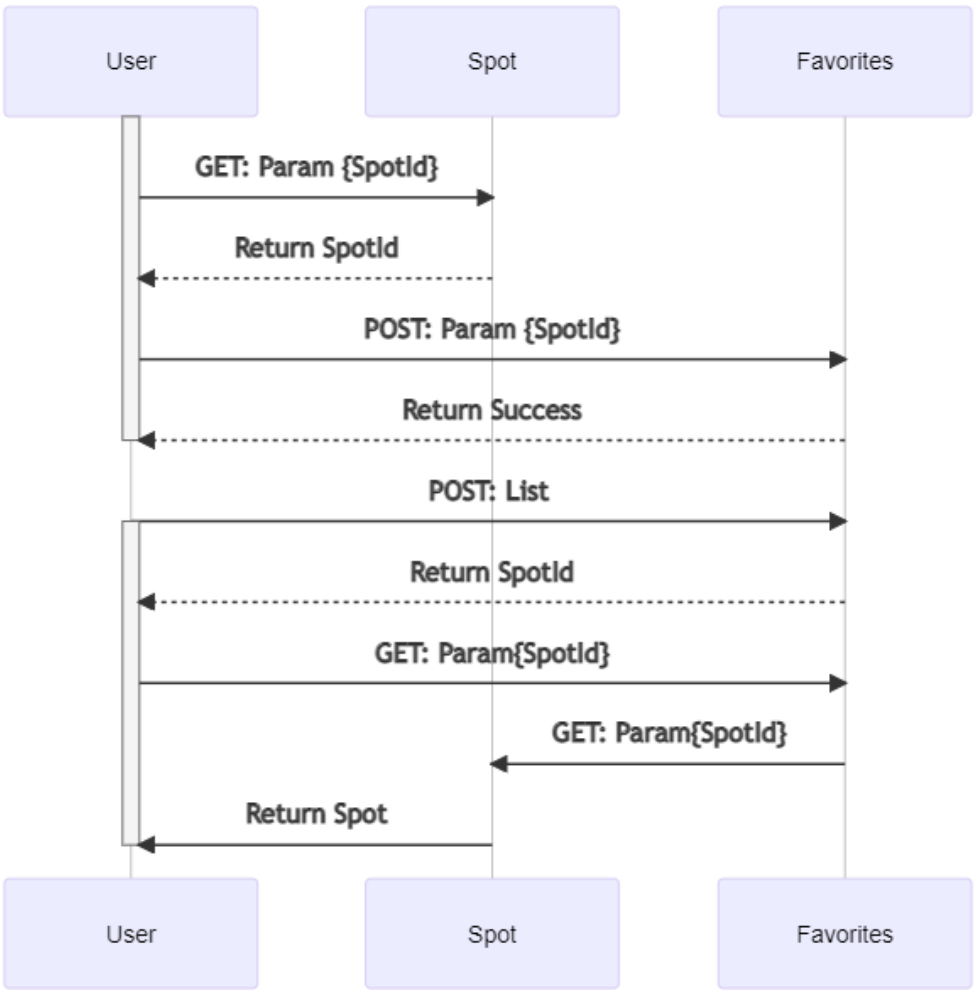

Figure 4-4: Message sequence diagram of Favorites

4.2.4 Functionality scenario: Review

Name: Peter makes a review and Maria views his review.

Initial State: Fisukortti web service is active. Peter is a registered user of Fisukortti and he wants to make a review for a spot. Maria is an unregistered user of Fisukortti.

Actors: Peter, Maria

Steps:

1. Peter reviews a spot.
2. Maria has opened a spot. She views reviews of that spot.

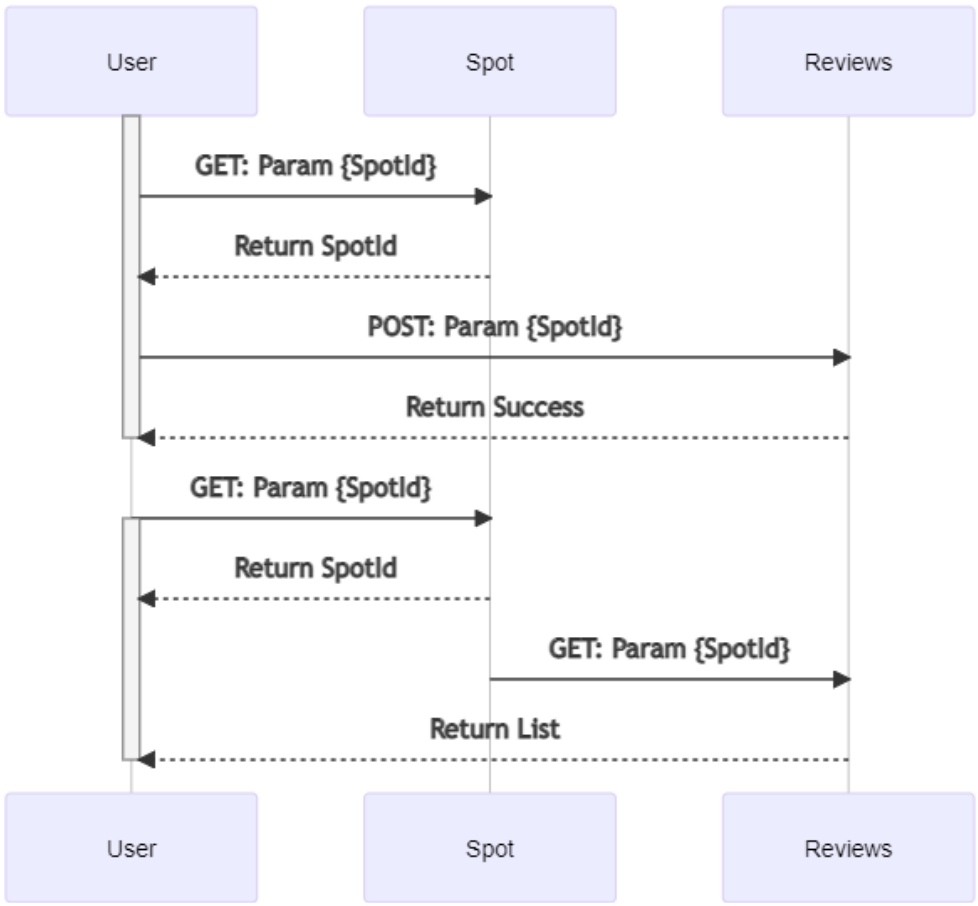

Figure 4-5: Message sequence diagram of Review

4.3 System context diagram

All client instances go through API Gateway.

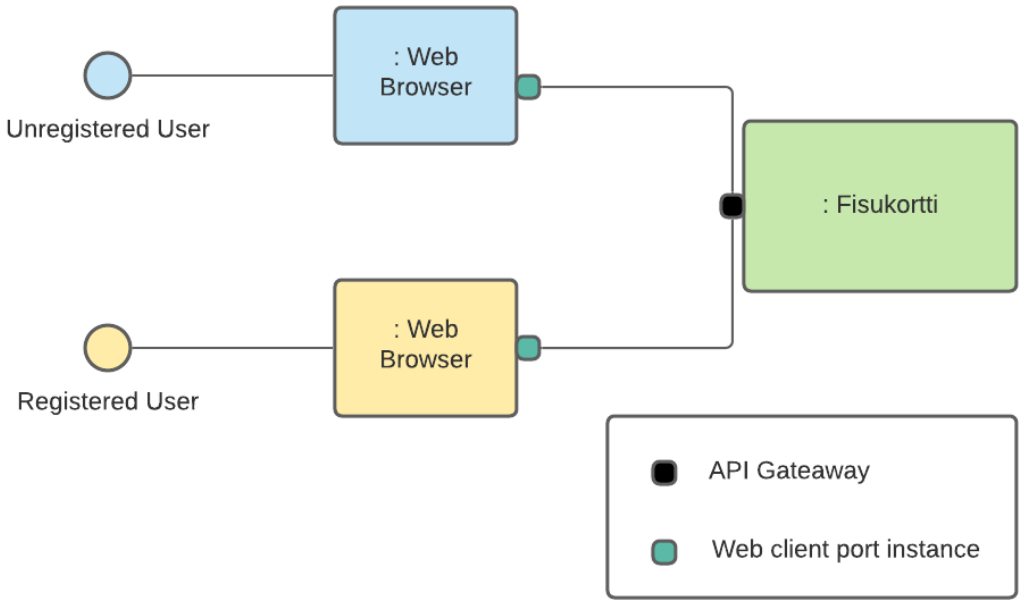

Figure 4-6: System context diagram

4.4 Top level component assembly diagram

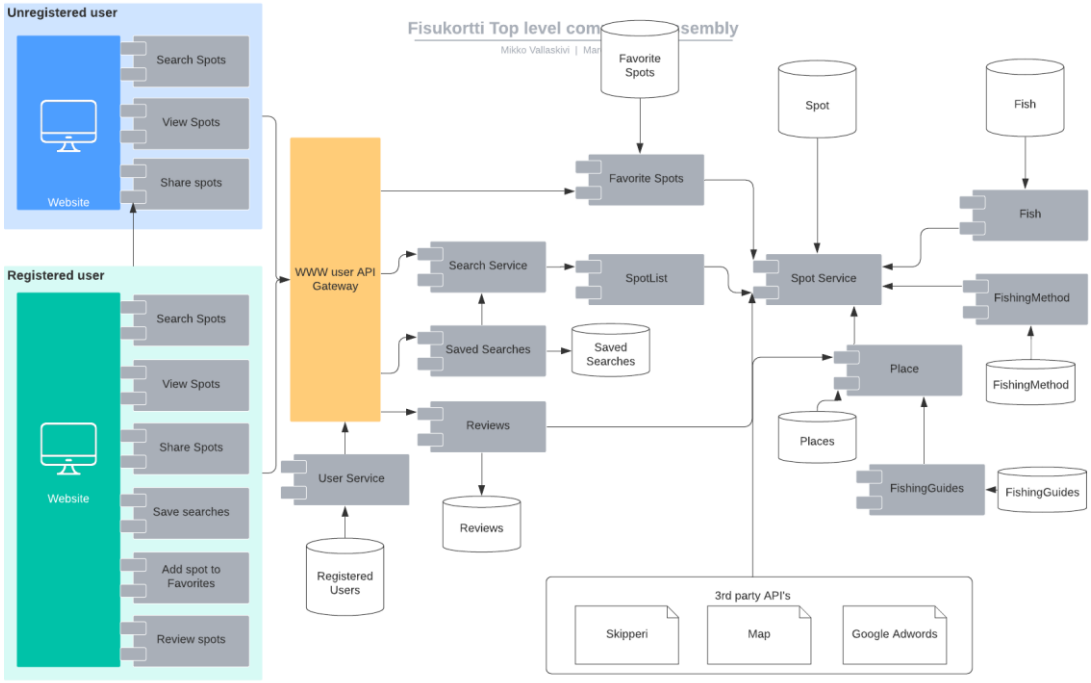

Figure 4-7 Component assembly diagram

Maintainability can be increased by using known patterns and by avoiding anti-patterns. Microlith is an anti-pattern where everything depends on everything else. This has been avoided in final design.

API gateway, that is the single entry point for all users, will be implemented. Different users need different data and API gateway provides the optimal API for each client. API gateway also simplifies the client by moving logic for calling multiple services from the client to API gateway. API gateway will increase complexity, but benefits are greater than drawbacks.

The API Gateway is key for quality attribute extensibility. By processing request through the API Gateway, support for new clients can be easily added, for example a native iOS application. The requests from the application will be processed to proper endpoints within the API Gateway, hence easing the server-side development.

The API Gateway authenticates the request and passes an access token that securely identifies the requestor in each request to the services. A service can include the access token in requests it makes to other services. Access token can handle authentication and authorization quality attributes.

4.5 Deployment diagram

Deployment diagram applies service deployment platform, serverless deployment and multiple service instances per host from Microservices pattern language.

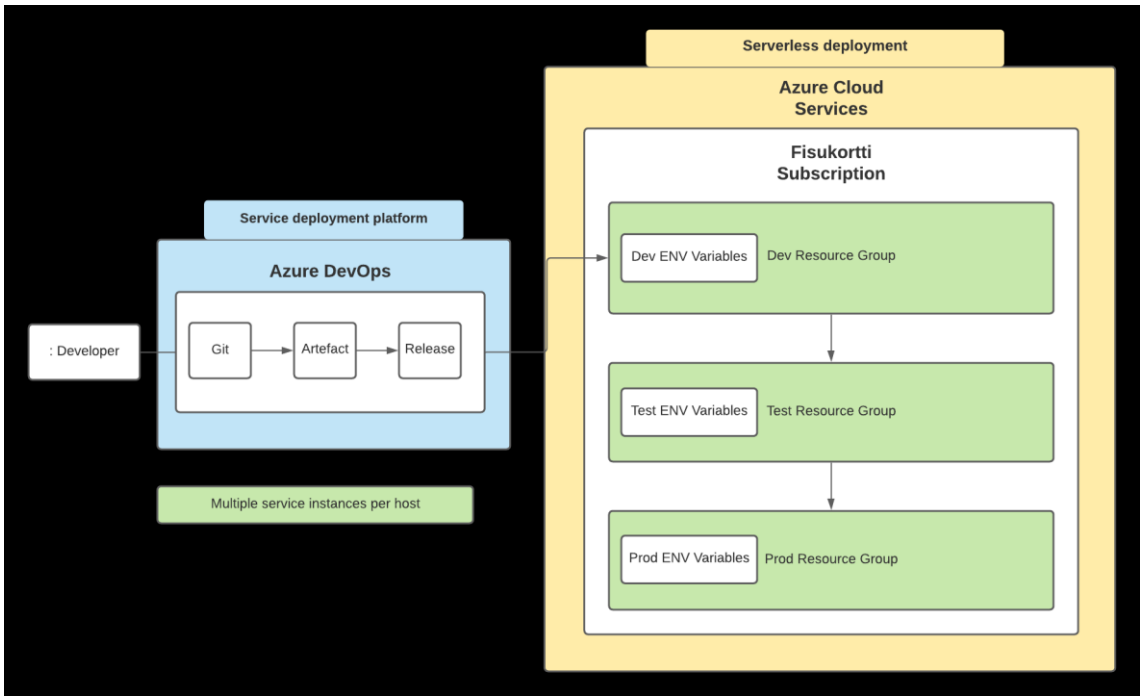

Figure 4-8: Deployment diagram

5 Assessment (Susanna and Mikko)

For assessment, four change scenarios were made.

5.1 Change scenario: new feature

Scenario: Adding new feature, AffiliateLinks, to the Fisukortti

Quality Attributes: Extensibility, maintainability

Environment: Normal operation

Stimulus: As popularity of the Fisukortti service rises, new affiliate partners want to do marketing through Fisukortti.

Response: New feature will be implemented, because it is profitable and will increase visibility of the Fisukortti.

Description of system behavior: AffiliateLinks will be displayed at the Spot page. New API calls and data processing needs to be implemented.

Argumentation: Scenario is highly relevant and easy to implement because of microservices architecture and well documented APIs provided by the affiliate partners.

5.2 Change scenario: changing the database

Scenario: Changing Fish-database so that Fisukortti will utilize Suomen lajitietokeskus's API in it.

Quality Attributes: Extensibility, maintainability

Environment: Normal operation

Stimulus: Suomen lajitietokeskus collects, shares, maintains information like pictures and sightings about species, and provides API so that information can be used.

Response: New feature might be implemented, because it would raise the quality of service in Fisukortti.

Description of system behavior: Adding 3rd party API to Fish microservice so that it will utilize Suomen lajitietokeskus's API. Fish database needs new columns for new information.

Argumentation: Scenario is relevant, but it will require quite a lot of work to implement.

5.3 Change scenario: assess feature

Scenario: Assessing feature SavedSearches

Quality Attributes: Usability, scalability

Environment: Normal operation

Stimulus: Caching search results would result in faster searches and less load on database queries. In addition, the cached results could be saved in browsers web cache so users could access the search results while disconnected from the internet.

Response: This feature should be built as a canary release to a small number of users who have been noticed in advance. This would allow us to test the usability of this feature and assess its need and impact on the service at larger scale. Redis cache needs to be implemented on the server.

Description of system behavior: After SpotList page is rendered to the user's browser, the results will be saved to web cache as well as Redis cache on the server.

Argumentation: Scenario is needed to test another way of sending the SpotList to the user, as when the service grows, the number of searches will increase which might have an impact on the databases.

5.4 Change scenario: new functionality

Scenario: Adding new functionality, search directly to the Spot service, Fisukortti

Quality Attributes: Usability

Environment: Normal operation

Stimulus: User can search a specific Spot by name querying directly the Spot database without rendering SpotList first.

Response: Service quality would increase as for users who want a specific spot, the search is a lot faster.

Description of system behavior: Add Spot table querying to Search, for example via autocompletion.

Argumentation: Scenario is highly relevant and will be easy to implement but will require a new endpoint in the Spot Service.

6 Figures