DOCUMENTATION

FACEBOOK ANALYTICS

ADVANCED WEB TECHNOLOGIES 2014/2015

Kostovski Ognen

814551

Lecturer: Prof. Piero Fraternali

Teaching assistant: Ilio Catallo

CONTENTS

FACEBOOK ANALYTICS INTRODUCTION

Facebook analytics application is built using JEE, adopting the Spring framework.

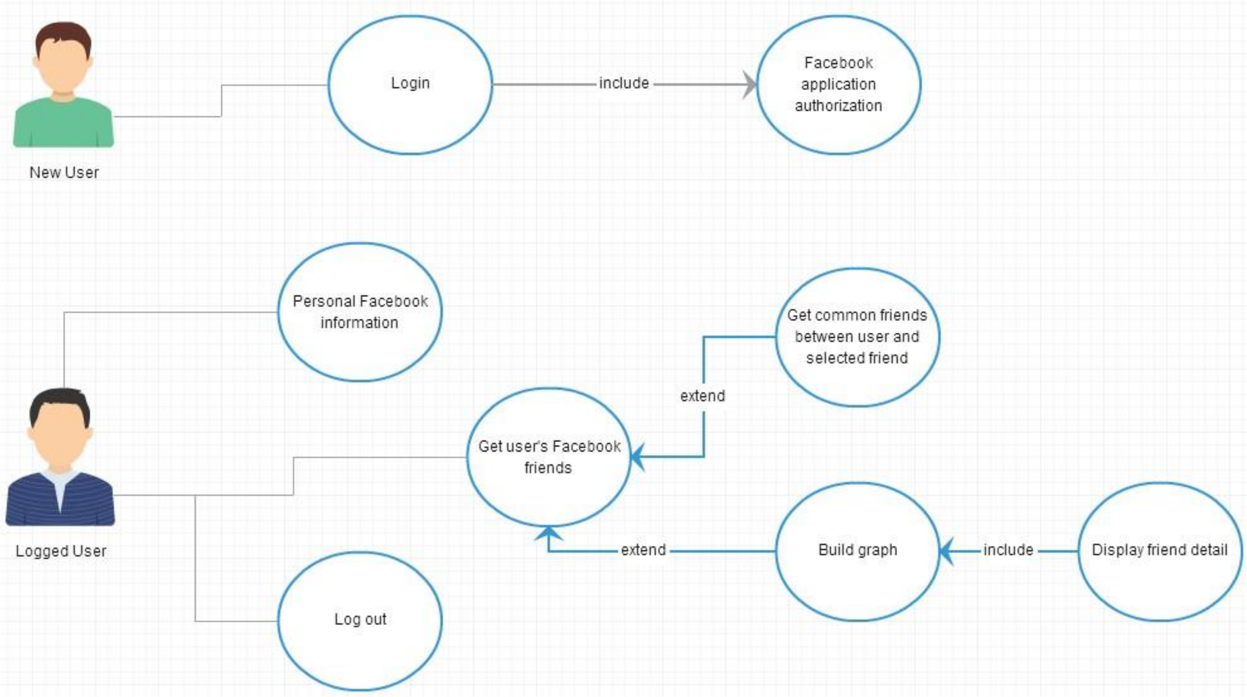
With the help of Facebook analytics we gather information about the users, their social network relationships. The application's business logic takes care of organizing this information in a way easily accessible, reusable and maintainable.

Spring MVC helps us with the application presentation, navigation and through the controller it allows us to use the implemented business logic.

The goal of this application is to design this system for acquiring, visualizing and metering the social relationships of people in a given social network (Facebook).

REQUIREMENTS ANALYSIS

EXTEND AND CLARIFY SPECIFIC ASPECTS OF THE PROJECT


New User
Login
include
Facebook application authorization
Personal Facebook information
Get common friends between user and selected friend
extend
Get user's Facebook friends
Logged User
extend
Build graph
include
Display friend detail
Log out

New User:

- Fill the login form in order to access web application’s services
- Authorize the application with the needed permissions

Logged User:

- Connected to the database
- Can navigate through the application and use its functionalities

DATA MODEL AND DATABASE SCHEMA

Facebook analytics has two database schemas:

- Friend: located in MySql database, configured via xml and managed through JPA
- Userconnection: located H2 in-memory database, configure through java

Friend:

This entity represents personal detail and relationships of a person (user, friend and common friend).

It is implemented as a POJO in it.polimi.awt.domain.Friend.java and persisted through hibernate.

Attributes:

- name: name and surname retrieved from social network profile
- objectId: auto-generated primary key
- facebookId: id retrieved from social network profile
- userId: current user facebookId, useful for realtionships
- loginDate: date of user log in
- searchDate: current date object from when the search was made
- graphMetrics: all metrics relevant to the graph
- parent (OneToMany): objectId for commonFriends
- commonFriends (ManyToOne): common friends of the user (= mutual friends with parent)

Relationships:

- Self-referencing relationship which links friends of the user with the common friends found through him/her.
- Implemented as a combination of two self-referencing relationships: parent (OneToMany) and commonFriends (ManyToOne)

Usage:

- Application users (userId = facebookId)
- Friends of user (userId = User.getFacebookId() and searchDate = null)
- Selected friends of user (userId = User.getFacebookId() and searchDate != null and parent = null)
- Friends in common (userId = User.getFacebookId() and searchDate != null and parent = Friend.getFacebookId())

ARCHITECTURE OF THE APPLICATION

DOMAIN MODEL

The POJO class Friend represents the core of the Facebook analytics domain model.

This class is used for representation of the user and any of his friends.

JPA is used to store the POJO and that is why we have getters and setters for all the attributes.

USER ACCESS HANDLING

User access handling is based on spring-social-quickstart example recommended by Spring Social.

For more information about the handling of user access please visit: http://projects.spring.io/spring-social-facebook/ where you can find the spring-social-quickstart example.

PRESENTATION LAYER

The presentation layer of the application is managed by the “Home Controller” class.

This class is based on business logic which operates with the service layer.

The service layer dependencies are injected and used in order to fill the model and return the proper view as a String.

SERVICE LAYER

List of services:

- **FriendsService**
  Communicates with persistence layer. PersistenceJPARepository object is autowired inside this service. The Data Access Object (DAO) is used to access the Database as a Service (DaaS).
  Functions: Add (User, Friends, Common friends), Find (Common friends)
- **UserFriend**
  Uses the retrieved Facebook object to create a Friend object.
- **FacebookFriends**
  Creates a List of Friend objects by using the facebook object and previously created User Friend object. The result is a List that represents Friend objects which are friends of the user.
  Function: Create Facebook Friends
- **CommonFriends**
  Creates a List of Friend objects by adding the Selected Friend object and adding all of his Facebook Friends (List of Friend objects) too.
  Result list is linked with Selected Friend as a parent !!
- **Graph**
  Create Graph function: Creates the graph for the user Friend object.
  Creates the graph instance, adds the user Friend object as first node, then all of his friends as nodes as edges. For each of his friends he creates nodes and edges between their common friends.
  Calculate Graph metrics function: From the created graph, the graph metrics are copied in the adequate metrics Friend object variables.
  Note: To render this graph on Client side we use associated json and javascript library “d3.js”
  Functions: Graph function, Graph metrics function
- **CreateJson**
  Creates JSON properly formatted for javascript library “d3.js”

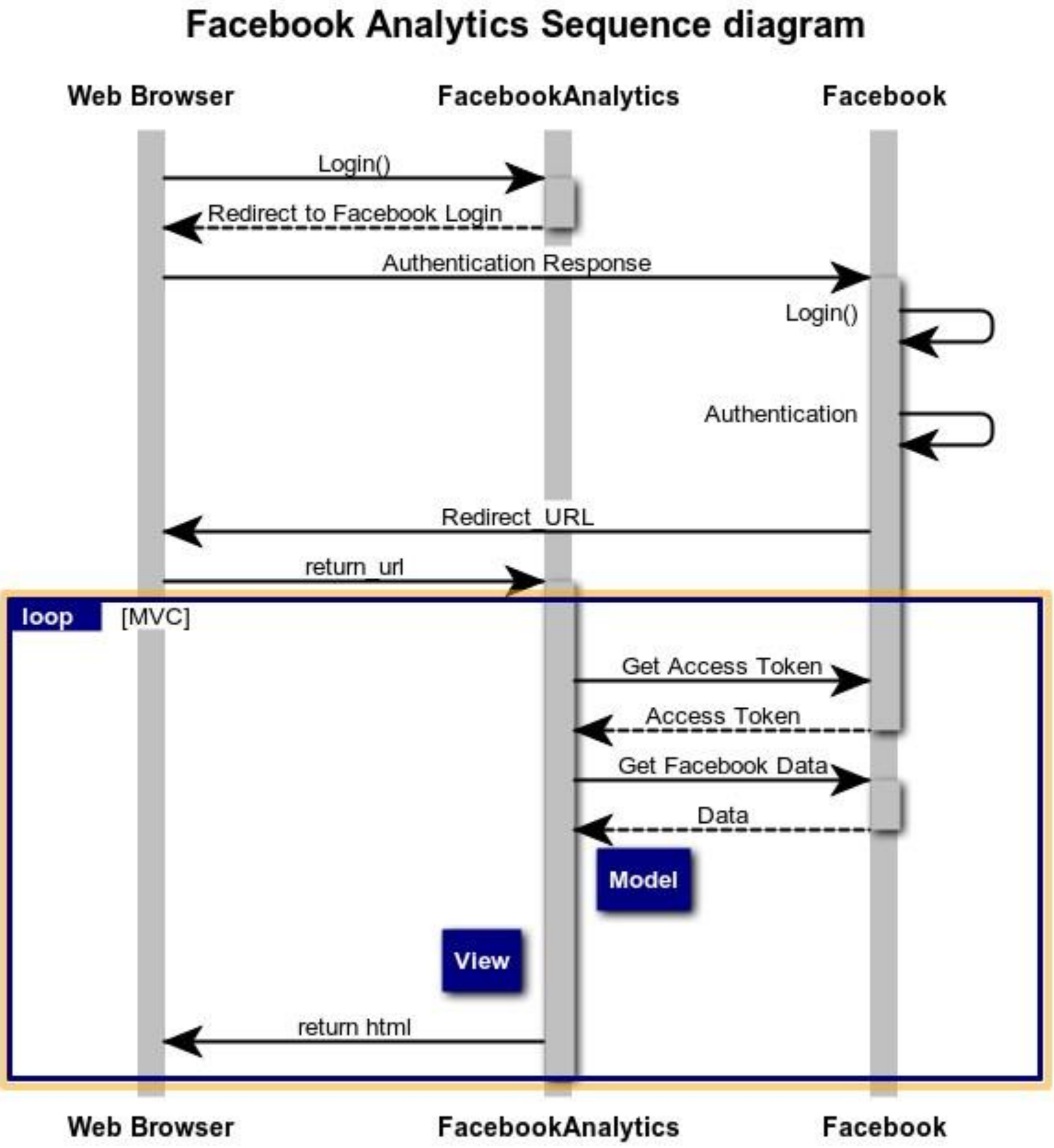
PERSISTENCE LAYER

FriendsJPARepository is the Data Access Object (DAO).

Here the entity manager is injected using “dataSource-context.xml” for the proper configurations so that, through hibernate, the POJO Friend can be persisted in the database.

Methods provided allow storing a Friend object or a List of Friend objects and searching for a List of Friend objects as a friend list for a specific Friend object.

FACEBOOK CONNECTION



CONCLUDING REMARKS

Facebook Analytics application is designed with 3-tier web architecture and uses Spring framework.

SPRING FRAMEWORK

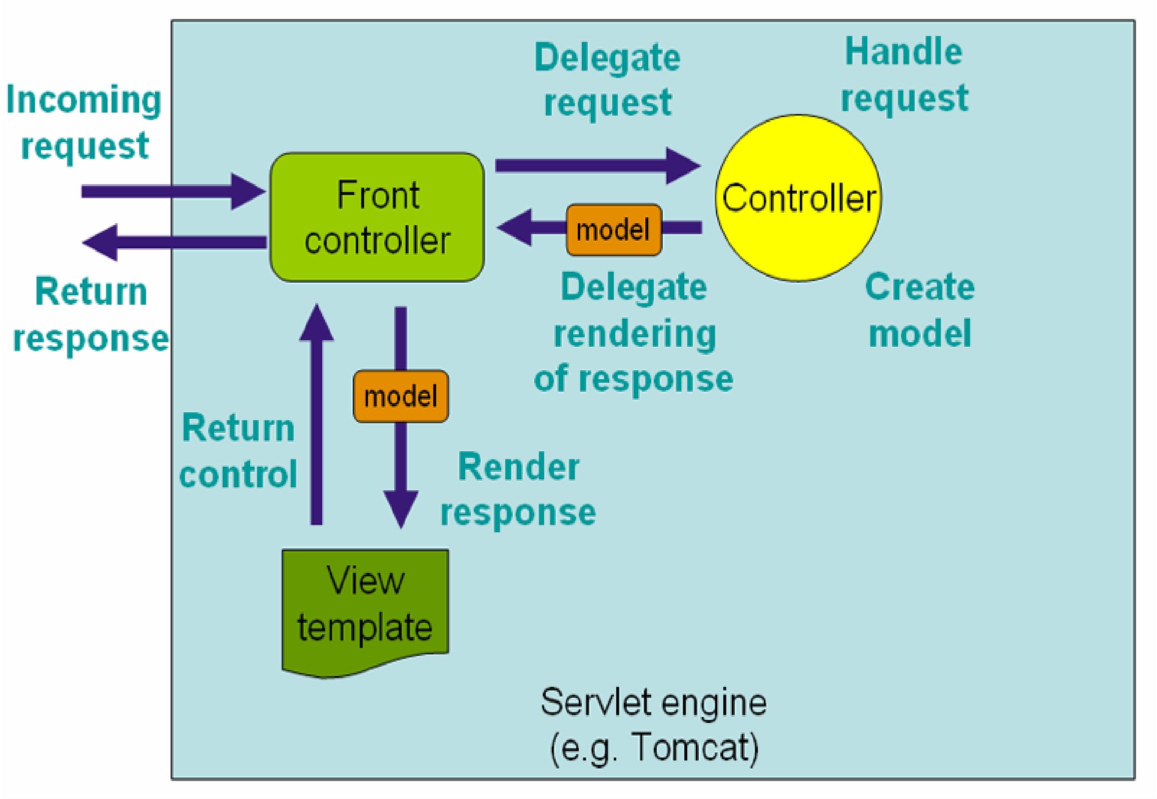
Spring Framework features:

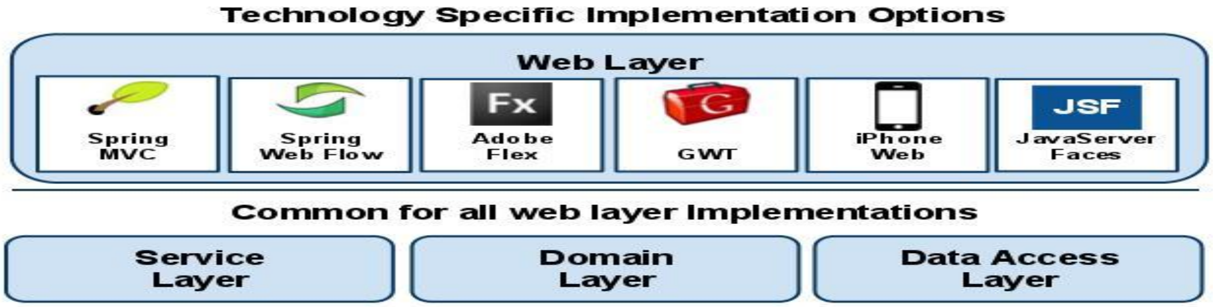
- Dependency Injection
- Aspect-Oriented programming
- Spring-MVC and RESTful framework
- Support JDBC, JPA, JMS

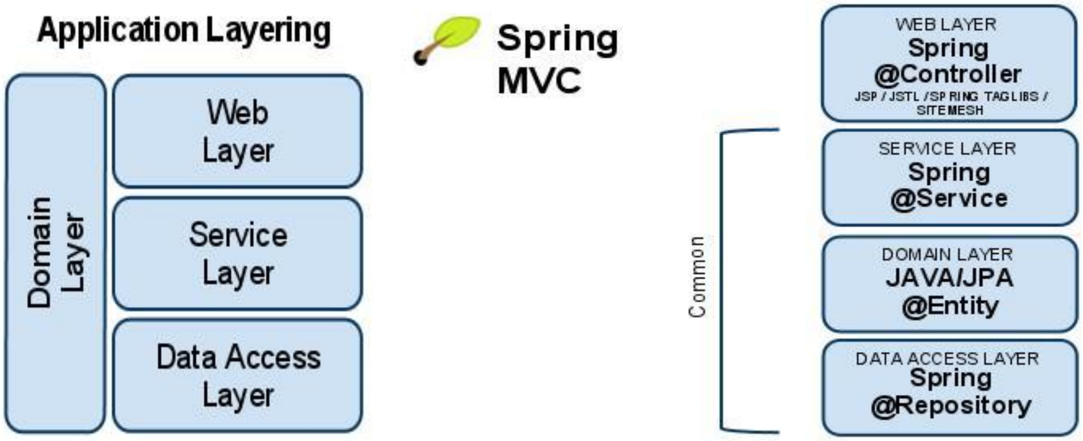
SPRING MVC

- Controller
- HandlerAdapter
- HandlerInterceptor
- HandlerMapping
- View
- ViewResolver

Spring version of MVC pattern:







SERVICE LAYER

Classes annotated with @Service, EJBs ready to be ejected.

Every bean is the implementation of an interface exposing the main functionalities.

DATA ACCESS LAYER

Persistence layer handled by hibernate (implementation of JPA).

Data Access Logic (DAO) managed through POJOs (intermediaries between application and database).

Separation of Persistence logic and business logic.

WEB LAYER

Views are created by combinations of: HTML, JSP, CSS, JSTL, Javascript.

Presentation logic:

- Server side (JSP, JSTL)
- Client side (HTMLL, CSS, Javascript)