Client

**Using one single instance of the application server or clustered? Microservices?**

One single instance(monolith) is used due to the simplicity of the program.

**What kind of client? Browser? Mobile phone? Java program on client computer?**

The main client is a browser but a prototype of a stand-alone client that communicates via web services shall be implemented as well. This is due to customer requirements.

The program must be compatible with:

Internet Explorer 11 and later

Firefox 50 and later

Chrome 55 and later

**Shall caching be used? What kind of caching? Browser, proxy, gateway?**

Shall be used if we choose the response time requirement.

Presentation layer

**How shall navigation be performed, what calls to the model should be made and which is the next view? Static, dynamic?**

The answers are to be written as a set of navigation rules.

**How shall validation of user input be made?**

Check if digits or text. If validation fails the same view should be displayed again, together with an error message explaining why the validation failed. Which validations and which error messages shall be defined as a set of validation rules.

**How shall flow control be used?**

Define a flow control scheme.

**How shall the view be separated into different parts? Header, footer, navigation…**

Header, footer, navigation, main content.

**Shall it be possible to switch language and add new languages? How?**

The application form of the app shall be available in different languages according to the requirements. It should be possible to add new languages.

**How are sessions managed?**

Sessions are needed for authentication.

**Which component is responsible for showing error messages?**

Bean validation.

**Which component is responsible for logging error messages?**

We use java util logging framework.

**What information shall be logged and what shall be shown?**

**How does information arrive to those components?**

**How shall authentication, authorization, privacy and integrity be handled? Servlet container?**

Servlet container.

Business layer

**Should the model retain state, that is should the result of one call still be present in the model when the next call is made?**

Language and authentication need to be stateful.

**When to synchronize with the database?**

Synchronization is done every call.

**What exceptions should be reported to presentation layer?**

Wrong username/password, existing user, transaction failed exception (when version number is incorrect).

**Which component is responsible for showing error messages? How does it arrive? What shall be logged?**

Bean validation

**How shall error messages be internationalized?**

By JSF internationalization

**Should presentation and business be in the same or separate processes?**

Same process but with web service in front of model.

**EJB calls local or remote?**

Local.

**How shall security be handled in the business layer?**

No need since the business layer is local and run in the same process as presentation.

**What data shall be used for communication with presentation? DTOs?**

DTOs!

**When should transactions start and stop?**

Transaction is needed for changing an application status.

**What transaction isolation should be used?**

A version number for the data is used.

**Do we need transactions when we read data? When we write data? When we read-update-write**

**data?**

Only when read-update-write.

**Which type of transaction propagation do we use?**

RequiresNew

Integration layer

**Shall it be possible to change DBMS without updating the business logic?**

No.

**How are relations, inheritance and other object oriented paradigms mapped to the database?**

JPA

**How are private keys generated and how do we ensure they are unique?**

JPA

**What data shall be used for communication with business? May the business logic objects themselves be used?**

DTOs

**How shall locking be used?**

Optimistic locking for updating application status.

**Shall lazy loading be used?**

Yes, unless the reference that is lazy loaded is being iterated.