Research application development frameworks that support JEE development, and develop a prototype using your chosen framework. (Note: if you have a particular interest in a non-java based framework, run it by me)

Technologies: As with number 6 above, there is a choice of technologies here that you could focus on, but will most likely involve JEE development and its supporting frameworks including JSF (front end) and JPA (data tier);

TITLE

Research application development frameworks that support JEE development, and develop a prototype using your chosen framework

EXECUTIVE SUMMARY

INTRODUCTION / BACKGROUND

Enterprise applications can consist of combinations of the following:

* Enterprise JavaBean ([EJB](https://en.wikipedia.org/wiki/EJB)) modules (packaged in [JAR files](https://en.wikipedia.org/wiki/JAR_file));
* Web modules (packaged in [WAR files](https://en.wikipedia.org/wiki/WAR_(Sun_file_format)));
* connector modules or [resource adapters](https://en.wikipedia.org/wiki/Resource_Adapter) (packaged in RAR files);
* [Session Initiation Protocol](https://en.wikipedia.org/wiki/Session_Initiation_Protocol) (SIP) modules (packaged in SAR files);
* application client modules;
* Additional JAR files containing dependent classes or other components required by the application;
* https://en.wikipedia.org/wiki/Java\_EE\_application

**Java EE**

The Java EE platform is built on top of the Java SE platform. The Java EE platform provides an API and runtime environment for developing and running large-scale, multi-tiered, scalable, reliable, and secure network applications[1].

https://docs.oracle.com/javaee/6/firstcup/doc/gkhoy.html

## Overview of Enterprise Applications

This section describes enterprise applications and how they are designed and developed.

As stated above, the Java EE platform is designed to help developers create large-scale, multi-tiered, scalable, reliable, and secure network applications. A shorthand name for such applications is “enterprise applications,” so called because these applications are designed to solve the problems encountered by large enterprises. Enterprise applications are not only useful for large corporations, agencies, and governments, however. The benefits of an enterprise application are helpful, even essential, for individual developers and small organizations in an increasingly networked world.

The features that make enterprise applications powerful, like security and reliability, often make these applications complex. The Java EE platform is designed to reduce the complexity of enterprise application development by providing a development model, API, and runtime environment that allows developers to concentrate on functionality.

### Tiered Applications

In a multi-tiered application, the functionality of the application is separated into isolated functional areas, called tiers. Typically, multi-tiered applications have a client tier, a middle tier, and a data tier (often called the enterprise information systems tier). The client tier consists of a client program that makes requests to the middle tier. The middle tier's business functions handle client requests and process application data, storing it in a permanent datastore in the data tier.

Java EE application development concentrates on the middle tier to make enterprise application management easier, more robust, and more secure.

#### The Client Tier

The client tier consists of application clients that access a Java EE server and that are usually located on a different machine from the server. The clients make requests to the server. The server processes the requests and returns a response back to the client. Many different types of applications can be Java EE clients, and they are not always, or even often Java applications. Clients can be a web browser, a standalone application, or other servers, and they run on a different machine from the Java EE server.

#### The Web Tier

The web tier consists of components that handle the interaction between clients and the business tier. Its primary tasks are the following:

* Dynamically generate content in various formats for the client.
* Collect input from users of the client interface and return appropriate results from the components in the business tier.
* Control the flow of screens or pages on the client.
* Maintain the state of data for a user's session.
* Perform some basic logic and hold some data temporarily in JavaBeans components.

##### Java EE Technologies Used in the Web Tier

The following Java EE technologies are used in the web tier in Java EE applications.

**Table 2-1 Web-Tier Java EE Technologies**

|  |  |
| --- | --- |
| **Technology** | **Purpose** |
| Servlets | Java programming language classes that dynamically process requests and construct responses, usually for HTML pages |
| JavaServer Faces technology | A user-interface component framework for web applications that allows you to include UI components (such as fields and buttons) on a page, convert and validate UI component data, save UI component data to server-side data stores, and maintain component state. |
| JavaServer Faces Facelets technology | Facelets applications are a type of JavaServer Faces applications that use XHTML pages rather than JSP pages. |
| Expression Language | A set of standard tags used in JSP and Facelets pages to refer to Java EE components. |
| JavaServer Pages (JSP) | Text-based documents that are compiled into servlets and define how dynamic content can be added to static pages, such as HTML pages. |
| JavaServer Pages Standard Tag Library | A tag library that encapsulates core functionality common to JSP pages |
| JavaBeans Components | Objects that act as temporary data stores for the pages of an application |

#### The Business Tier

The business tier consists of components that provide the business logic for an application. Business logic is code that provides functionality to a particular business domain, like the financial industry, or an e-commerce site. In a properly designed enterprise application, the core functionality exists in the business tier components.

##### Java EE Technologies Used in the Business Tier

The following Java EE technologies are used in the business tier in Java EE applications:

**Table 2-2 Business Tier Java EE Technologies**

|  |  |
| --- | --- |
| **Technology** | **Description** |
| Enterprise JavaBeans (enterprise bean) components | Enterprise beans are managed components that encapsulate the core functionality of an application. |
| JAX-RS RESTful web services | An API for creating web services that respond to HTTP methods (for example GET or POST methods). JAX-RS web services are developed according to the principles of REST, or representational state transfer. |
| JAX-WS web service endpoints | An API for creating and consuming SOAP web services. |
| Java Persistence API entities | An API for accessing data in underlying data stores and mapping that data to Java programming language objects. |
| Java EE managed beans | Managed components that may provide the business logic of an application, but do not require the transactional or security features of enterprise beans. |

#### The Enterprise Information Systems Tier

The enterprise information systems (EIS) tier consists of database servers, enterprise resource planning systems, and other legacy data sources, like mainframes. These resources typically are located on a separate machine than the Java EE server, and are accessed by components on the business tier.

##### Java EE Technologies Used in the EIS Tier

The following Java EE technologies are used to access the EIS tier in Java EE applications:

**Table 2-3 EIS Tier Java EE Technologies**

|  |  |
| --- | --- |
| **Technology** | **Description** |
| The Java Database Connectivity API (JDBC) | A low-level API for accessing and retrieving data from underlying data stores. A common use of JDBC is to make SQL queries on a particular database. |
| The Java Persistence API | An API for accessing data in underlying data stores and mapping that data to Java programming language objects. The Java Persistence API is a much higher-level API than JDBC, and hides the complexity of JDBC from the user. |
| The Java EE Connector Architecture | An API for connecting to other enterprise resources, like enterprise resource planning or customer management system software. |
| The Java Transaction API (JTA) | An API for defining and managing transactions, including distributed transactions or transactions that cross multiple underlying data sources. |

<https://docs.oracle.com/javaee/6/firstcup/doc/gcrky.html>

An application framework is a collection of classes implementing the shared architecture of a family of applications. A technique is proposed for defining the specialization interface of a framework in such a way that it can be used to automatically produce a task-driven programming environment for guiding the application development process. Using the environment, the application developer can incrementally construct an application that follows the conventions implied by the framework architecture. The environment provides specialization instructions adapting automatically to the application-specific context, and an integrated source code editor which responds to actions that conflict with the given specialization interface. The main characteristics and implementation principles of the tool are explained.

<http://link.springer.com/chapter/10.1007/3-540-44800-4_15>

[Generating Application Development Environments for Java Frameworks

* Markku Hakala
* , Juha Hautamäki
* , Kai Koskimies
* , Jukka Paakki
* , Antti Viljamaa
* , Jukka Viljamaa]

What are application frameworks? Why are they important to you? What do they provide that can help you get your job done?

A framework’s primary purpose is to aid and ease your application development process. It should allow you to develop the application quickly and easily and should result in a superior finished application [2].

Book java frameworks and components pg12

<http://struts.apache.org/struts1eol-press.html>[3]

It is important that you see the real benefit to determine whether mounting the learning curve of a framework is worthwhile.

Frameworks, in brief, provide you with a powerful *tool box*. The tools in this box help in many different areas of application development. They provide essential design patterns and structure to your application development project, and also provide the backbone and container for the components you create for your application to operate within. In Chapter 2 we will explore the kinds of services and tools you will commonly find in frameworks.

Frameworks are valuable at all stages of development, from design to de- ployment and beyond, perhaps more so in ongoing maintenance. They usually apply to almost all stages of the life cycle of an application.

1.2.3.1 **COMPONENTS AND APPLICATION FRAMEWORKS**

A framework must be as simple as possible, but no simpler. In other words, unnecessary complexity should not get in the way, and yet there must be enough capability provided that the framework has real and measurable benefit.

1.2.3.2 **COMPONENTS AND APPLICATION FRAMEWORKS**

MAIN RESEARCH QUESTION

What are frameworks?

What frameworks exist for Java Enterprise Application development?

What are core features of the most used frameworks?

Where should these frameworks be implemented in order to have most efficient impact?

Can I build a prototype in one of the frameworks researched?

JUSTIFICATION / BENEFITS

Components come in many different forms, from the ubiquitous JavaBean to Web services, and at many different levels of scale. At the small-scale level, components with a visual aspect, such as drop-down boxes, tree lists, and buttons, are assembled into a single form. At the high-scale level, entire services such as order entry, billing, and inventory update are combined into a complete application.

All levels of components, however, have one thing in common: they all operate within some specific component container. Often, this container is one of the services provided by a framework.

FEASABILITY

Who I am

My ability

My workstation

Learning curve

Timeline

Resources access to college library, supervisor Geraldine Gray

METHODS

Compare data size of framework classes, Lightweight / heavy and Removal of Framework

Impact using framework on project code

Learning curve of frameworks reviewed

How frameworks manage dependency injection

Testing for resource consumption / computational time JVM

EXPECTED RESULTS

Initial difficult learning curve

Lots of frameworks of different sizes for different needs

Frameworks ease development process

Result in superior finished application

The ability to develop a prototype using a Framework

CONCLUSION

REFERENCES

PROJECT PLAN