|  |
| --- |
| **Concordia University**  **Department of Computer Science**  **and Software Engineering** |

|  |
| --- |
| **F.S.T.S.**  ***Family Services Tracking System*** |

|  |
| --- |
| **SAD** |

|  |
| --- |
| **SOEN 390**  **Software Development Project**  **Winter 2012** |

**JAM PACK**

Katrina Anderson

Cynthia Donato

Joshua Hum

Mikhael Levkovsky

Adrian Lloyd

Patrick Modafferi

**Table of Contents**

[1. Introduction 7](#_Toc321782328)

[1.1 Purpose 7](#_Toc321782329)

[1.2 Scope 7](#_Toc321782330)

[1.3 Definitions, Acronyms and Abbreviations 8](#_Toc321782331)

[1.4 References 8](#_Toc321782332)

[2. Architectural Representation 8](#_Toc321782333)

[3. Architectural Goals and Constraints 9](#_Toc321782334)

[3.1 MVC vs 3-Tier Architecture 9](#_Toc321782335)

[4. Use Case View 10](#_Toc321782336)

[4.1 Actor Goal List 10](#_Toc321782337)

[4.2 Use Case Model 12](#_Toc321782338)

[4.2.1 Administrator Use Case Models 12](#_Toc321782339)

[4.2.2 Employee Use Case Models 14](#_Toc321782340)

[4.3 Fully Dressed Use Cases 16](#_Toc321782341)

[UC1.1 CreateClient File 16](#_Toc321782342)

[UC1.2 Update Client File 17](#_Toc321782343)

[UC1.3 Set Client File Inactive 18](#_Toc321782344)

[UC1.4 Add Note to Client File 19](#_Toc321782345)

[UC1.5 Add Flag to Client File 20](#_Toc321782346)

[~~UC1.6 Archive Client Files~~ 21](#_Toc321782347)

[UC1.7 Search for a Client File by Client File ID 22](#_Toc321782348)

[UC1.8 Search for a Client File Using Advanced Search 23](#_Toc321782349)

[UC1.9 Create a Dependent on a Client File 24](#_Toc321782350)

[UC1.10 Delete a Dependent on a Client File 25](#_Toc321782351)

[UC1.11 Update a Dependent on a Client File 26](#_Toc321782352)

[UC1.12 Remove Flag From a Client File 27](#_Toc321782353)

[UC1.13 Create Flag 28](#_Toc321782354)

[UC1.14 Set Flag Inactive 29](#_Toc321782355)

[UC2.1 Create Event Template 30](#_Toc321782356)

[UC2.2 Update Event Template 31](#_Toc321782357)

[UC2.3 Delete Event Template 32](#_Toc321782358)

[UC2.4 View All Event Templates 33](#_Toc321782359)

[UC2.5 View Event Template Details 34](#_Toc321782360)

[UC2.6Create Event Occurrence 35](#_Toc321782361)

[UC2.7 Create Category 36](#_Toc321782362)

[UC2.8 Update Event Occurrence 37](#_Toc321782363)

[UC2.9 Cancel Event Occurrence 38](#_Toc321782364)

[UC2.10 View All Event Occurrences 39](#_Toc321782365)

[UC2.11 View Event Occurrence Details 40](#_Toc321782366)

[UC3.1 Create Appointment 41](#_Toc321782367)

[UC3.2 Update Appointment 42](#_Toc321782368)

[UC3.3 Cancel Appointment 43](#_Toc321782369)

[UC3.4 View All Booked Appointments 44](#_Toc321782370)

[UC4.1 Get All Event Attendees 45](#_Toc321782371)

[UC4.2 Appointment Fulfillment 46](#_Toc321782372)

[UC4.3 Print Attendee List 48](#_Toc321782373)

[~~UC5.1 ManageReport Templates~~ 49](#_Toc321782374)

[UC5.2 Generate Operational Reports 50](#_Toc321782375)

[UC5.3 Generate Statistical Reports 51](#_Toc321782376)

[5. Logical View 52](#_Toc321782377)

[5.1 Operational Contracts 52](#_Toc321782378)

[CO1.1 Register a Client 52](#_Toc321782379)

[CO1.2 Register a Dependent 53](#_Toc321782380)

[CO1.3 Deregister a Dependent 53](#_Toc321782381)

[CO1.4 Create Flag 53](#_Toc321782382)

[CO1.5 Remove Flag 54](#_Toc321782383)

[CO1.6 Add Note 54](#_Toc321782384)

[CO2.1 Create Event Template 54](#_Toc321782385)

[CO2.2 Delete Event Template 55](#_Toc321782386)

[CO2.3 Create Event Occurrence 55](#_Toc321782387)

[CO2.4 Delete Event Occurrence 55](#_Toc321782388)

[CO3.1 Book an Appointment 56](#_Toc321782389)

[CO3.2 Cancel an Appointment 56](#_Toc321782390)

[5.2 Domain Model 57](#_Toc321782391)

[5.3 Class Diagram 58](#_Toc321782392)

[5.4 System Sequence Diagrams 63](#_Toc321782393)

[SSD1.1 Register a Client 63](#_Toc321782394)

[SSD1.2 Register a Dependent 63](#_Toc321782395)

[SSD1.3 Deregister a Dependent 63](#_Toc321782396)

[SSD1.4 Create Flag 64](#_Toc321782397)

[SSD1.5 Remove Flag 64](#_Toc321782398)

[SSD1.6 Add Note 64](#_Toc321782399)

[SSD2.1 Create Event 65](#_Toc321782400)

[SSD2.2 Delete Event 65](#_Toc321782401)

[SSD2.3 Create Event Occurrence 65](#_Toc321782402)

[SSD2.4 Delete Event Occurrence 66](#_Toc321782403)

[SSD3.1 Book Appointment 66](#_Toc321782404)

[SSD3.2 Cancel Appointment 66](#_Toc321782405)

[5.5 Sequence Diagram 67](#_Toc321782406)

[5.6 Communication Diagrams 67](#_Toc321782407)

[CD1.1 Register a Client 67](#_Toc321782408)

[CD1.2 Register a Dependent 68](#_Toc321782409)

[CD1.4 Create a Flag 68](#_Toc321782410)

[CD1.6 Add a Note 68](#_Toc321782411)

[CD2.1 Create Event Template 69](#_Toc321782412)

[CD2.3 Create Event Occurrence 69](#_Toc321782413)

[CD3.1 Book an Appointment 69](#_Toc321782414)

[6. Development View 70](#_Toc321782415)

[6.1 Package Diagram 70](#_Toc321782416)

[6.2 Component Diagram 73](#_Toc321782417)

[7. Process View 74](#_Toc321782418)

[7.1 Activity Diagrams 74](#_Toc321782419)

[AD1.1 Register a Client 74](#_Toc321782420)

[AD1.2 Register a Dependent 75](#_Toc321782421)

[AD1.3 Deregister a Dependent 76](#_Toc321782422)

[AD1.4 Create Flag 77](#_Toc321782423)

[AD1.5 Remove a Flag 78](#_Toc321782424)

[AD1.6 Add a Note 79](#_Toc321782425)

[AD2.1 Create Event Template 80](#_Toc321782426)

[AD2.2 Delete Event Template 81](#_Toc321782427)

[AD2.3 Create Event Occurrence 82](#_Toc321782428)

[AD2.4 Delete Event Occurrence 83](#_Toc321782429)

[AD3.1 Book an Appointment 84](#_Toc321782430)

[AD3.2 Cancel an Appointment 85](#_Toc321782431)

[8. Physical View 86](#_Toc321782432)

[8.1 Deployment Diagram 86](#_Toc321782433)

[9. Yii Framework Diagrams 87](#_Toc321782434)

[9.1 Yii Static Structure 87](#_Toc321782435)

[9.2 Workflow Graph 87](#_Toc321782436)

[10. Size & Performance 88](#_Toc321782437)

[11. Quality 88](#_Toc321782438)

[Appendix A References 89](#_Toc321782439)

|  |
| --- |
| **F.S.T.S.** |

|  |
| --- |
| **SAD** |

|  |
| --- |
| **Version 6.25** |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Rev.** | **Description** | **Author(s)** |
| 2012-01-14 | 0.0 | Contributed sections 2 and 3, User Goal List and Use Case Model respectively. | Katrina Anderson |
| 2012-01-14 | 0.1 | Contributed section 4 Fully Dressed Use Cases | Cynthia Donato |
| 2012-01-15 | 0.2 | Contributed and edited fully dressed use cases from other teams. | Cynthia Donato & Katrina Anderson |
| 2012-01-15 | 0.3 | Domain Model | Mikhael Levkovsky & Josh Hum |
| 2012-01-15 | 0.4 | Updated Use Case Model | Mikhael Levkovsky |
| 2012-01-15 | 0.5 | Updated Actor Goal List | Adrian Lloyd |
| 2012-01-15 | 0.6 | Added SRS links, formatted and reviewed document | Katrina Anderson |
| 2012-01-25 | 0.7 | Yii Framework Diagrams | Mikhael Levkovsky |
| 2012-01-29 | 1.8 | Modified other product requirements | Mikhail Levkovsky |
| 2012-01-29 | 1.9 | Contributed additional Use Cases and modified Use case model and actor goal list. | Cynthia Donato |
| 2012-01-30 | 1.10 | Revised Document | Katrina Anderson |
| 2012-02-08 | 2.11 | Contributed use cases and updated use case model and actor goal list. | Cynthia Donato |
| 2012-02-13 | 2.12 | Revised Document | Katrina Anderson |
| 2012-02-13 | 2.13 | Sequence Diagram | Mikhael Levkovsky |
| 2012-02-26 | 3.14 | Revised Document | Katrina Anderson |
| 2012-02-26 | 4.15 | Revised Document | Katrina Anderson |
| 2012-03-09 | 4.16 | Added Operational Contracts | Katrina Anderson |
| 2013-03-11 | 4.17 | Added Architecture Diagrams | Mikhael Levkovsky |
| 2012-03-20 | 5.18 | Document Refactoring | Katrina Anderson |
| 2012-03-20 | 5.19 | Added Use cases 3.5 and 5.3 | Adrian Lloyd |
| 2012-03-21 | 5.20 | Started Initial Architecture Sections 1, 2 & 3 | Adrian Loyd |
| 2012-03-22 | 5.21 | Updated Sections 1, 2, 3, 10 and 11 | Mikhail Levkovsky |
| 2012-03-22 | 5.22 | Update use cases. Remove 3.5 update 3.1 | Cynthia Donato |
| 2012-03-22 | 5.23 | Reviewed Sections 1, 2, 3, 10 and 11 | Adrian Lloyd |
| 2012-03-25 | 5.24 | Added Graphs and Revised Document | Katrina Anderson |
| 2012-04-09 | 6.25 | Updated Document | Katrina Anderson |

# Introduction

When the development of F.S.T.S. was being initially planned, we decided that to deliver the best product possible we would develop F.S.T.S. as a web application. In order to be able to develop at a fast pace while still maintaining a high development standard we decided to adopt a framework. After some research we decided on the Yii PHP framework. This framework ensured an MVC style architecture as well as providing a high level of automation when it came to development. The Yii framework drove our development and allowed us to build the system we currently have.

## Purpose

The purpose of this document is to define the main components of F.S.T.S. with architectural artifacts. With these artifacts we can determine the flow of the system, how components interact with one another and how the system is constructed and deployed as a whole.

## Scope

The scope of the architecture artifacts cover the main components of the system:

|  |  |
| --- | --- |
| Use Case View: | * Actor-Goal List * Use Case Diagrams * Use Cases |
| Logical View: | * Operational Contracts * Domain Model * Class Diagram * System Sequence Diagrams * Sequence Diagrams * Communication Diagrams |
| Development View: | * Package Diagram * Component Diagram |
| Process View: | * Activity Diagrams |
| Physical View: | * Deployment Diagram |
| Other: | * Yii Framework Diagrams |

The two main components of our system have been narrowed down to:

* Client Files
* Events

Therefore the artifacts defined in this document are linked to the 2 main components defined above.

## Definitions, Acronyms and Abbreviations

Please see section 14-Glossary of the Supplementary Specification document.

## References

Please see Appendix A, References, of this document.

# Architectural Representation

The architecture of this system is the Model View Controller. The Yii framework enforces this architecture and decouples each component into their own respective folders for the component’s model, view and controller.

The architecture has multiple views representing it.The Use-Case View, Logical View, Development View, Process View and the Physical View all give a more complete understanding of the system.

The Use Case view was derived directly from the requirements (mostly the functional ones). This view broke down all the functionalities of the system and gives us insight into exactly what the system should do. This allowed us to create our traceability matrices and keep track of what functionality was completed. This also allowed to keep track of what our users wanted and how to achieve each functionality.

The Logical View contains our Communication Diagrams (see Section 5.6) and our System Sequence Diagrams (see Section 5.4). The Communication Diagrams gives us insight into how entities communicate with each other, what relationships and associations get created and passed across our system when different components need to pass messages between each other. The System Sequence Diagrams treats F.S.T.S. as a black box. It illustrates what information a user needs to provide to the system and what the system will give back as output depending on the use case.

The Development View contains our Package Diagrams (see Section 6.1) and our Component Diagram (see Section 6.2). The Package Diagram represents how our files and classes are logically separated. It also shows how the different components in the packages interact with each other, as well as the dependancies between them. The Component Diagram gives us an overall image of all the major parts of the F.S.T.S. system. It demonstrates how the user interacts with our system, and which messages get passed from one compoenent to another. The Component Diagram also gives us insight into what each component contains.

The Process View contains our Activity Diagrams (see Section 7.1). The Activity diagram shows a stepwise workflow of all the activities and actions that occur when a user tries to complete a user story. It deals with concurrency and explains the order of activities that occur for an activity to complete.

The Physical View contains our Deployment Diagram (see Section 8.1). The Deployement Diagram shows the physical entities that together make up F.S.T.S. and how to connect to it.This explains that to use our system one would need either a Generic PC, an iPad or an iPhone. Requests are passed from the user to the server machine via HTTP requests. The server then communicates with a Database which in itself can be a physical entity.

These diagrams were derived either using our use cases and our operational contracts, or by the technical constraints imposed on us by the working environment at Welcome Hall Mission.

# Architectural Goals and Constraints

The reason for choosing the MVC architecture is to achieve:

* Decouple presentation, data and behaviour
* Allows for constant maintenance
* Allows for multiple people to work on the same project
* Allows for low-coupling between different components
* High cohesion
* Promotes organization and code reuse

Adopting the Yii framework allows for faster development as well as easier maintance after deployment. By adopting a well known framework that enforces an architectural pattern, such as MVC, will help any future developer maintain our product. Since we are using the Yii framework, it should be easy for any future developer to understand our structure and make changes as necessary. All the documentation can be found online <http://www.yiiframework.com/>

Some Contraints of the MVC architecture are:

* Learning curve if the MVC architecture is unfamiliar
* Complexity may increase due to auxillary patterns that are co-occure with MVC
* Increased coupling between the model, view and controller of a single component (Please note the difference between this point and point 4 above) since a change in the model has to propagated to both the view and the controller.

## MVC vs 3-Tier Architecture

A few drawbacks of the 3-Tier Architecture is that in a 3-Tier Architecture the Client Tier can’t talk to to the Data Tier. In our web application this is not necessarily a restriction we want to impose on ourselves. While the view mostly reflects the model, it still contains some basic logic and the Data Tier needs to be queried in rare instances. Also, a 3 Tier Architecture usually has its layers separated via a network. However, in our small production environment that is Welcome Hall Mission everything will be hosted on one machine (or at most two).

In this case MVC seemed to be a perfect fit. It is very popular for Web application, where the View is rendered via XHTML/CSS, the Controller receives the input via the GET or POST methods, and the Model is represented by our Domain objects.

# Use Case View

## Actor Goal List

|  |  |
| --- | --- |
| **Actor** | **Goal** |
| Administrator | Archive Client Files  Create Event Template  Update Event Template  Delete Event Template  View All Event Templates  View Event Template Details  Create Income Type  Update Income Type  Delete Income Type  Create Income Period  Update Income Period  Delete Income Period  Create Work Status  Update Work Status  Delete Work Status  Create Postal Code Mapping  Update Postal Code Mapping  Delete Postal Code Mapping  Create User  Update User  Delete User  Create Language  Update Language  Delete Language  Create Country  Update Country  Delete Country  Create Referral  Update Referral  Delete Referral  Create Flag  Update Flag  Delete Flag  Set flag Inactive  Create Citizenship Status  Update Citizenship Status  Delete Citizenship Status  Create Marital Status  Update Marital Status  Delete Marital Status  Create Relationship  Update relationship  Delete Relationship |
| Employee | Create Client File  Update Client File  Set Client File Inactive  Add Note to Client File  Update Note to Client File  Delete Note from Client File  Add Flag to Client File  Remove Flag on a Client File  Client Search  Create household member  Delete household member  Update household member  Add Income  Update Income  Delete Income  Create Event Occurrence  Update Event Occurrence  Delete Event Occurrence  View all Event Occurrences  View event Occurrence Details  Create Appointment  Cancel an Appointment  View All Booked Appointments  Get All Event Attendees  Appointment Fulfillment  Print Attendee List |

## Use Case Model

### Administrator Use Case Models





### Employee Use Case Models





## Fully Dressed Use Cases

### UC1.1 CreateClient File

ID:UC1.1

Use Case:Create Client File

Description:Employee wishes to create a family‘s file in the F.S.T.S. system.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Family: Their interest is to get confirmation by the Employee that their file is up-to-date with the needed information.  
Employee: Their interest is to make sure updates to the family files have been properly registered in the system.  
Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

N/A

Post Conditions:

Success end condition

1. Family file is updated in the F.S.T.S. Client Files System

Failure end condition:

1. Client file remains unchanged
2. Employee is notified that the system failed to add family

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee navigates to the client file management section
2. System displays the client file management options
3. Employee indicates that they would like to create a new client file
4. System requests that client information be provided
5. Employee specifies client information and initiates file creation
6. System notifies employee that client file has been created

Extensions:

5.a Client residence already exists in the system

5.a.1 System notifies the user that the residence is already in use by another client

5.a.2 System provide options for conflict resolution

5.a.2.1 System provides the option to remove the residence from the original client file

5.a.2.2 System provides the option to create client and dependents of the new file as dependents on the original client file

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.2 Update Client File

ID:UC1.2

Use Case:Update Client File

Description:Employee wishes to update a family‘s file in the F.S.T.S. system.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Family: Their interest is to get confirmation by the Employee that their file is up-to-date with the needed information.  
Employee: Their interest is to make sure updates to the family files have been properly registered in the system.  
Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. The client file to be updated exists in the system

Post Conditions:

Success end condition

1. Family file is updated in the F.S.T.S. Client Files System

Failure end condition:

1. Client file remains unchanged
2. Employee is notified that the system failed to update the family file.

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee initiates client file search
2. System returns client file
3. Employee updates file information
4. System notifies employee that client file has been updated

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.3 Set Client File Inactive

ID:UC1.3

Use Case:Set Client File Inactive

Description:Employee wishes to set a family‘s file in the F.S.T.S. system to an inactive status.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Employee: Their interest is to make sure the family files are up to date in the system.

Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. The client file to be set inactive exists in the system

Post Conditions:

Success end condition

1. Family file is set inactive in the F.S.T.S. Client Files System

Failure end condition:

1. Client file remains unchanged
2. Employee is notified that the system failed to set the family inactive

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee initiates client file search
2. System returns client file
3. Employee indicates they would like to set the file inactive
4. System notifies employee that client file has been set to inactive mode

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.4 Add Note to Client File

ID:UC1.4

Use Case:Add note to client file

Description:Employee wishes add a note to a family‘s file in the F.S.T.S. system.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Client: Their interests are to be served quickly and efficiently by an employee who is aware of specific facts about their family.

Employee: Their interest is to make sure the family files are descriptive and easy to understand for other employees viewing the file.

Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. The client file exists in the system.

Post Conditions:

Success end condition

1. A note is added to the family file containing the author and date/time the note was written.

Failure end condition:

1. Client file remains unchanged.
2. Employee is notified that the system failed to add the note to the client file.

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee initiates client file search.
2. System returns client file.
3. Employee indicates they would like to add a note.
4. System displays an area for adding a detailed note.
5. Employee enters the detailed note.
6. Employee initiates the addition of the note to the file.
7. System notifies employee that the note has been added to the client file.

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.5 Add Flag to Client File

ID:UC1.5

Use Case:Add flag to a client flag

Description:Employee wishes add a flag to a family‘s file in the F.S.T.S. system.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Employee: Their interest is to make sure that important information is easily visible in the client file.

Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. The client file exists in the system.
2. At least one flag has been configured in the system.

Post Conditions:

Success end condition

1. A flag is added to the family file.

Failure end condition:

1. Client file remains unchanged.
2. Employee is notified that the system failed to add the flag to the client file.

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee initiates client file search.
2. System returns client file.
3. Employee indicates they would like to add a flag.
4. System displays a list of predefined flags.
5. Employee selects the flag they would like to add from the list.
6. System notifies employee that the note has been added to the client file.

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### ~~UC1.6 Archive Client Files~~

~~ID: UC1.6~~

~~Use Case:Archive Client Files~~

~~Description:Employee wishes archive all client files who have been inactive for a specified period of time.~~

~~Level:User-goal~~

~~Primary Actor:Administrator~~

~~Supporting Actors~~**~~:~~** ~~None~~

~~Stakeholders and Interests:~~

~~Administrator: Keep the system clean and running fast.~~

~~Employee: Their interest is to have a quick responsive search.~~

~~Developer: Their interest is to make sure that updates to client filesfunction properly.~~

~~Pre-Conditions:~~

~~1. Client files must exist in F.S.T.S. and be in an inactive state.~~

~~2. Client files which are must have been set inactive for over a specified time period~~

~~Post Conditions:~~

~~Success end condition~~

1. All client files that have been inactive within the specified time period are archived

~~Failure end condition:~~

~~1. No client files are archived~~

~~2. An error message is displayed to the user~~

~~Minimal Guarantee~~

1. ~~All active client files and client files which are not within the specified time period are unaffected~~

~~Main Success Scenario:~~

1. ~~Employee navigates to the administrator section of the website~~
2. ~~System displays a list of available operations~~
3. ~~Employee indicates that they would like to archive client files~~
4. ~~System prompts user for the minimum inactivity period~~
5. ~~Employee enters the minimum inactivity period~~
6. ~~System displays list of families who have been inactive for longer than that period~~
7. ~~Employee indicates which families are to be archived~~
8. ~~System indicates that all selected families have been archived.~~

~~Extensions:~~

~~6.aSystem warns employee that the entered minimum inactivity period is not valid~~

~~Special Requirements:~~

~~See Section 2 Functionalityof Supplementary Specifications document~~

~~See Section 3 Usabilityof Supplementary Specifications document~~

~~See Section 4 Reliabilityof Supplementary Specifications document~~

~~See Section 5 Performanceof Supplementary Specifications document~~

~~See Section 6 Supportabilityof Supplementary Specifications document~~

~~See Section 7 Design Constraintsof Supplementary Specifications document~~

~~See Section 10 Interfacesof Supplementary Specifications document~~

\*This feature has been scoped our by the client.

### UC1.7 Search for a Client File by Client File ID

ID:UC1.7

Use Case:Search For a Client File By Client File ID

Description:Employee wishes to view a client file while in posession of the client file ID.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Family: Their interest is to have their file located quickly.  
Employee: Their interest is to locate the client file quickly.  
Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. Employee is logged into the system
2. The client file id exists in the system

Post Conditions:

Success end condition

1. 1. Family file is displayed by the F.S.T.S. Client Files System

Failure end condition:

1. 1. Client file is not found
2. 2. Employee is notified that the system failed to find the family’s file

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee navigates to the client file search
2. System prompts the user to enter the client file Id they wish to locate
3. Employee enters the client file Id
4. System returns the client file associated to the client file Id

Extensions:

4.a Client Id not found in system

4.a.1 System notifies the user that the client file is not found

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.8 Search for a Client File Using Advanced Search

ID:UC1.8

Use Case:Search For a Client File Using Advanced Search

Description:Employee wishes to view a client file by search using family representative’s name, dependents name or residence.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Family: Their interest is to have their file located quickly.  
Employee: Their interest is to locate the client file quickly.  
Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. Employee is logged into the system
2. The client file id exists in the system

Post Conditions:

Success end condition

1. Family file is displayed by the F.S.T.S. Client Files System

Failure end condition:

1. Client file is not found
2. Employee is notified that the system failed to find the family’s file

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee navigates to the client file search
2. System prompts the user to enter the client file Id they wish to locate or try searching using the advanced search
3. Employee indicates they would like to use the Advanced Search
4. System prompts the user to enter any of the following: family representative’s name, dependents name or address of residence.
5. Employee any one of the requested criteria
6. System returns a list of client files that meet the provided criteria
7. Employee selects a client file from the list
8. System displays the details of the client file

Extensions:

10.a No Client files meet the entered search criteria

10.a.1 System notifies the user that no client files meet those criteria

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.9 Create a Dependent on a Client File

ID:UC1.9

Use Case:Create a dependent on a client file

Description:Employee opens the client file and indicates that they would like to add a dependent. The employee fills in all required information adds the dependent.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Employee: Their interest is to make sure that client files stay up-to-date.

Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. Employee is authorized and authenticated.
2. The client file exists in the system.

Post Conditions:

Success end condition

1. A dependent is added to the client file.

Failure end condition:

1. The client file is unchanged.
2. Employee is notified that the system failed to add the dependent to the client file.

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee initiates client file search.
2. System returns client file.
3. Employee indicates they would like to add a dependent to the file.
4. System requests the dependent first and last name, Medicare number and type. It is also possible to specify mother tongue, language of communication and country of origin.
5. System notifies employee that the dependent was added to the client file.

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.10 Delete a Dependent on a Client File

ID:UC1.10

Use Case:Delete a dependent on a client file

Description:Employee opens the client file, chooses a dependent from the list and indicates that they would like to remove the record.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Employee: Their interest is to make sure that client files stay up-to-date.

Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. Employee is authorized and authenticated.
2. The client file exists in the system.
3. Dependent is already added to the client file.

Post Conditions:

Success end condition

1. A dependent is removed from the client file.

Failure end condition:

1. The client file is unchanged.
2. Employee is notified that the system failed to add the dependent to the client file.

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee initiates client file search.
2. System returns client file.
3. Employee selects a dependent from the list of current dependents and indicates that they would like to remove the dependent.
4. System prompts the user to confirm that the dependent will be permanently removed from the client file.
5. Employee confirms that the dependent should be removed.
6. System notifies employee that the dependent was removed from the client file.

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.11 Update a Dependent on a Client File

ID:UC1.11

Use Case:Update a dependent on a client file

Description:Employee opens the client file, chooses a dependent from the list and indicates that they would like to update the record.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Employee: Their interest is to make sure that client files stay up-to-date.

Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. Employee is authorized and authenticated.
2. The client file exists in the system.
3. Dependent is already added to the client file.

Post Conditions:

Success end condition

1. A dependent is updated with the new information.

Failure end condition:

1. The client file is unchanged.
2. Employee is notified that the system failed to update the dependent on the client file.

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee initiates client file search.
2. System returns client file.
3. Employee selects a dependent from the list of current dependents and indicates that they would like to update the dependent.
4. System displays the dependent information in a updatable form.
5. Employee makes necessary updates and indicated they would like to save the changes.
6. System notifies employee that the dependent was updated from the client file.

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.12 Remove Flag From a Client File

ID:UC1.12

Use Case:Remove flag from a client file

Description:Employee opens up a client file and indicates that they would like to remove a flag.

Level:User-goal

Primary Actor:Employee

Secondary Actor: None

Stakeholders and Interests:

Employee: Their interest is to make sure that client files stay up-to-date.

Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. The client file exists in the system.
2. At least one flag has been configured in the system.
3. The flag has been added to the client flag.

Post Conditions:

Success end condition

1. A flag is removed from the family file.

Failure end condition:

1. The flag is not removed.
2. Employee is notified that the system failed to remove the flag from the client file.

Minimal Guarantee

1. An informative message is displayed to the user

Main Success Scenario:

1. Employee initiates client file search.
2. System returns client file.
3. Employee indicates they would like to removeone of the set flags on the file.
4. System notifies employee that the note has been removed from the client file.

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.13 Create Flag

ID: UC6.1

Use Case:Create Flag

Description:Administrator navigates to the Administrator section and indicates they would like to create a new flag.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Interest is to create a new flag for employees to quickly add to client files.

Pre-Conditions:

1. Administrator is identified and authenticated.

Post Conditions:

Success end condition

1. The flag is created.

Failure end condition:

1. A flag is not created in the system.
2. An error message is displayed to the user.

Minimal Guarantee

1. An informative message is displayed to the user.

Main Success Scenario:

1. Administrator navigates to the Administrator section.
2. System displays a list of available operations.
3. Administrator indicates that they would like to create a new flag.
4. System requests the name and description of the new flag.
5. Administrator specifies a name and description for the flag.
6. Administratorchooses to create the flag.
7. System indicates that the flag has been created.

Extensions:

7.a Mandatory information is missing

7.a.1 Error message indicating that the user has omitted mandatory information is displayed

7.a.2 Event template is not deleted

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC1.14 Set Flag Inactive

ID: UC6.2

Use Case:Set Flag Inactive

Description:Administrator navigates to the Administrator section and indicates they would like to set an existing flag inactive.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Interest is to set a flag inactive so that employees can no longer add it to client files.

Pre-Conditions:

1. Administrator is identified and authenticated.

Post Conditions:

Success end condition

* 1. The flag is set inactive.

Failure end condition:

1.The flag is not set inactive in the system.

2. An error message is displayed to the user.

Minimal Guarantee

* + 1. An informative message is displayed to the user.

Main Success Scenario:

1. Administrator navigates to the Administrator section.
2. System displays a list of available operations.
3. Administrator indicates that they would like to set a flag inactive.
4. System displays a list of active flags.
5. Administrator chooses one or more flags to set inactive.
6. Administratorinitiates the action of setting the selected flags inactive.
7. System indicates that the flag or flags have been set inactive.

Extensions:

N/A

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.1 Create Event Template

ID: UC2.1

Use Case:Create Event Template

Description:Administrator navigates to the Event Template Management section and creates a new event template.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Wants to create an event template in order to facilitate event occurrence creation.

Pre-Conditions:

1. Administrator is identified and authenticated

Post Conditions:

Success end condition

1. The new event template is created.

Failure end condition:

1. A new event template is not created in the system
2. An error message is displayed to the user

Minimal Guarantee

1. All previously created events remain unaffected

Main Success Scenario:

1. Administrator navigates to the Event Template Management section.
2. System displays a list of available operations.
3. Administrator indicates that they would like to create a new event template.
4. System displays event template creation form.
5. Administrator enters the name of the event, specifies if appointments can be created for this event, specifies if the event is repeatable and if so on which weekdays.
6. Administrator indicates they would like to create the event.
7. System indicates that the event has been created.

Extensions:

6.a Mandatory information is missing

6.a.1 Error message indicating that the user has omitted mandatory information is displayed

6.a.2 Event template is not created

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.2 Update Event Template

ID: UC2.2

Use Case:UpdateEvent Template

Description:Administrator navigates to the Event Template Management section and updates an event template.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Wants to update an event template in order to facilitate event occurrence creation.

Pre-Conditions:

1. Administrator is identified and authenticated.

Post Conditions:

Success end condition

1. The event template is updated.

Failure end condition:

1. A event template is not updated in the system.
2. An error message is displayed to the user.

Minimal Guarantee

1. An informative message is displayed to the user.

Main Success Scenario:

1. Administrator navigates to the Event Template Management section.
2. System displays a list of available operations.
3. Administrator indicates that they would like to update a particular event template.
4. System displays a list of existing event template information.
5. Administrator indicates they would like to update a specific event from the list.
6. System displays the details of the event template.
7. Administratorupdates any of the following fields: name of the event, if appointments can be created for this event, if the event is repeatable and if so on which weekdays.
8. Administrator indicates they would like to save the changes.
9. System indicates that the event has been saved.

Extensions:

9.a Mandatory information is missing

9.a.1 Error message indicating that the user has omitted mandatory information is displayed

9.a.2 Event template is not updated

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.3 Delete Event Template

ID: UC2.2

Use Case:DeleteEvent Template

Description:Administrator navigates to the Event Template Management section and indicates they would like to delete and existing template.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Wants to update an event template in order to facilitate event occurrence creation.

Pre-Conditions:

1. Administrator is identified and authenticated.

Post Conditions:

Success end condition

1. The event template is deleted.

Failure end condition:

1. A event template is not deleted from the system.
2. An error message is displayed to the user.

Minimal Guarantee

1. An informative message is displayed to the user.

Main Success Scenario:

1. Administrator navigates to the Event Template Management section.
2. System displays a list of available operations.
3. Administrator indicates that they would like to delete a particular event template.
4. System displays a list of existing event templates.
5. Administratorchooses to delete an event template from the list.
6. System indicates that the event has been deleted.

Extensions:

6.a Mandatory information is missing

6.a.1 Error message indicating that the user has omitted mandatory information is displayed

6.a.2 Event template is not deleted

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.4 View All Event Templates

ID: UC2.4

Use Case:View All Event Templates

Description:Administrator would like to view a list of all event templates.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Wants to view all event templates that are available in the system.

Pre-Conditions:

1. Administrator is identified and authenticated.

Post Conditions:

Success end condition

1.All event templates are displayed.

Failure end condition:

1.No templates are displayed.

2. An error message is displayed to the user.

Minimal Guarantee

* 1. An informative message is displayed to the user.

Main Success Scenario:

1. Administrator navigates to the Event Template Management section.
2. System displays a list of existing event templates.

Extensions:

2.a No Event templates exist

2.a.1 Error message indicating that no event templates were found is displayed

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.5 View Event Template Details

ID: UC2.5

Use Case:View Event Template Details

Description:Administrator would like to view the details of an event template.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Wants to view event template details for a particular template.

Pre-Conditions:

1. Administrator is identified and authenticated.
2. An event template exists in the system

Post Conditions:

Success end condition

1.Event template details are display.

Failure end condition:

* 1. An error message is displayed to the user.

Minimal Guarantee

* 1. An informative message is displayed to the user.

Main Success Scenario:

1. Administrator navigates to the Event Template Management section.
2. System displays a list of existing event templates.
3. Administrator indicates they would like to view the details of a particular template
4. System displays the details of the event template

Extensions:

N/A

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.6Create Event Occurrence

ID: UC2.6

Use Case:Create Event Occurrence

Description:Employee logs into the system creates a new event.

Level:User-goal

Primary Actor:Employee

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Makes sure individual event information is gathered.

Developer: Their interest is to make sure that updates to client filesfunction properly.

Pre-Conditions:

1. Employee is identified and authenticated

Post Conditions:

Success end condition

* 1. The new event is created.

Failure end condition:

1.A new event is not created in the system

2. An error message is displayed to the user

Minimal Guarantee

1.All previously created events remain unaffected

Main Success Scenario:

1. Employeenavigates to the Event section
2. System displays a list of available operations
3. Employee indicates that they would like to create a new event
4. System displays event creation form
5. Employee enters the name, date and specifies if appointments can be created for this event and saves.
6. System indicates that the event has been created

Extensions:

6.a Mandatory information is missing

6.a.1Error message indicating that the user has omitted mandatory information is displayed

6.a.2 Event is not created

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.7 Create Category

ID: UC2.7

Use Case:Create category

Description:Administrator logs into system and would like to create a new category for dependents.

Level:User-goal

Primary Actor:Administrator

Supporting Actors:None

Stakeholders and Interests:

Employee: Have a quick way to add a dependent type to a file.

Pre-Conditions:

Administrator is identified and authenticated

Post Conditions:

Success end condition

1.A new dependents category is created

Failure end condition:

1.A new category is not created

2. An error message is displayed to the client

Minimal Guarantee

1.All existing categories are unaffected.

Main Success Scenario:

1. Administrator navigates to the administrator section
2. System displays a list of available operations
3. Administrator indicates that they would like to create a new dependents category
4. System displays the category creation form
5. Administrator enters the new category name and saves the information
6. System confirms that a new category was created

Extensions:

6.a Category name is not specified

6.a.1 Error message is displayed indicating that the category name was not specified

6.a.2The category is not created.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.8 Update Event Occurrence

ID: UC2.8

Use Case:Update Event Occurrence

Description:Administrator logs into the system and updates either the max capacity for the event of the date of the occurrence.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Makes sure individual event information accurate.

Developer: Their interest is to make sure that event occurrences stay up to date so that appointments can be properly created.

Pre-Conditions:

1. Administrator is identified and authenticated
2. The event occurrence to be updated exists in the system and is active

Post Conditions:

Success end condition

* 1. The event occurrence is updated.

Failure end condition:

* + 1. The event occurrence is not updated in the system
    2. An error message is displayed to the user

Minimal Guarantee

1.A descriptive message is displayed to the user

Main Success Scenario:

1. Administratornavigates to the EventManagement area of the Administrator section.
2. System displays a list of active events.
3. Administrator selects the appropriate event from a list and indicates they would like to update it.
4. System the event details in a modifiable form.
5. Administrator makes the necessary updates and indicates he/she would like to save the changes.
6. System indicates that the event occurrence has been updated.

Extensions:

2.a No events are currently open

2.a.1 Message indicating that no events are active is displayed

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.9 Cancel Event Occurrence

ID: UC2.9

Use Case:Cancel Event Occurrence

Description:Administrator logs into the system and cancels a particular event occurrence.

Level:User-goal

Primary Actor:Administrator

Supporting Actors**:** None

Stakeholders and Interests:

Administrator: Makes sure individual event information is gathered.

Developer: Their interest is to make sure that updates to event occurrences also cancel all the individual appointments for that event.

Pre-Conditions:

1. Administrator is identified and authenticated
2. The event occurrence to be cancelled exists in the system and is active
3. There are no active appointments for the event occurrence

Post Conditions:

Success end condition

* 1. The event occurrence is cancelled.
  2. All existing appointments for the event are cancelled.
  3. A printable list of all cancelled appointments is displayed to the administrator.

Failure end condition:

* + 1. The event occurrence is not cancelled in the system

1. All individual appointments for the event occurrence remain intact
2. An error message is displayed to the user

Minimal Guarantee

1.A descriptive message is displayed to the user

Main Success Scenario:

1. Administratornavigates to the EventManagement area of the Administrator section.
2. System displays a list of active events.
3. Administrator selects the appropriate event from a list and indicates they would like to cancel it.
4. System displays a warning message explaining that all appointments for the event will be removed.
5. Administrator acknowledges the warning.
6. System indicates that the event occurrence has been cancelled and displays a list of all appointments that were cancelled along with the phone number of the family so that they may be alerted.

Extensions:

2.a No events are currently open

2.a.1 Message indicating that no events are active is displayed

5.a Administrator decides not to cancel the event

5.a.1 The event occurrence is not cancelled

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

### UC2.10 **View All Event Occurrences**

ID: UC2.10

Use Case:View All Event Occurrences

Description:Employees would like to view a list of all event occurrences.

Level:User-goal

Primary Actor:Employee

Supporting Actors**:** None

Stakeholders and Interests:

Employee: Wants to view all event occurrences that are active in the system.

Pre-Conditions:

1. Employee is identified and authenticated.

Post Conditions:

Success end condition

1. All event occurrences are displayed.

Failure end condition:

1. No event occurrences are displayed.
2. An error message is displayed to the user.

Minimal Guarantee

1. An informative message is displayed to the user.

Main Success Scenario:

1. Employee navigates to the Active Eventssection.
2. System displays a list of existing event occurrences, their dates, the max capacity and the number of currently registered patrons.

Extensions:

2.a No Event occurrences exist

2.a.1 Error message indicating that no events were found is displayed

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC2.11 **View Event Occurrence Details**

ID: UC2.11

Use Case:View Event Occurrence Details

Description:Employee would like to view the details of an event to ensure that appointments can be appropriately made.

Level:User-goal

Primary Actor:Employee

Supporting Actors**:** None

Stakeholders and Interests:

Employee: Wants to view up-to-date event details for a particular occurrence.

Pre-Conditions:

1. Employee is identified and authenticated.
2. An event occurrence exists in the system

Post Conditions:

Success end condition

1.Event details are display.

Failure end condition:

* 1. An error message is displayed to the user.

Minimal Guarantee

* 1. An informative message is displayed to the user.

Main Success Scenario:

1. Employeenavigates to the Active Eventssection.
2. System displays a list of existing events.
3. Employee selects the event they would like to view.
4. System displays the details of the event.

Extensions:

N/A

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC3.1 Create Appointment

ID: UC3.1

Use Case: Create Appointment

Description:The employee receives a phone call from the client and starts the application. The user searches for the clients file and creates an appointment for the respective event.

Level:User Goal

Primary Actor:Employee

Supporting Actors:N/A

Stakeholders and Interests:

Welcome Hall Mission: Keep events organized by creating appointments for attendees.

Client: Diminish wait time by booking an appointment for an event.

Employee: Keep crowds of attendees manageable and ensure that no over booking occurs.

Concordia University Students: Produce a system that functions as expected by the Mission.

Pre-Conditions:

1. Client has phoned for an appointment
2. Client has a file with the Welcome Hall Mission
3. The event exists in the system

Post Conditions:

Success end condition:

1. An appointment is created
2. The appointment is associated to the client file

Failure end condition:

1. Appointment has not been created

Minimal Guarantee:

1. All existing appointments are unaffected
2. A descriptive message is displayed to the user.

Main Success Scenario:

1. Employee initiates client file search
2. System returns client file
3. Employee ensures that client is eligible for the respective appointment
4. Employee initiates the make appointment functionality
5. System prompts the user to specify the event, date and time for the appointment
6. Employee provides the requested criteria
7. Employee submits the appointment
8. System confirms the appointment creation

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

8.1 Event has reached max capacity for all timeslots

8.1.1 Alert is displayed to the user informing them that max capacity is reached

8.1.2 Employee indicates that they would like to add the client to the waiting list

8.1.3 The system confirms that the client has been added to the waiting list

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

### UC3.2 Update Appointment

ID: UC3.2

Use Case: Update Appointment

Description:The employee receives a phone call from the client and starts the application. The user searches for the clients file and views their existing appointments. The employee selects the appointment to be updated and makes the update.

Level:User Goal

Primary Actor:Employee

Supporting Actors:N/A

Stakeholders and Interests:

Welcome Hall Mission: Keep events organized by creating appointments for attendees.

Client: Diminish wait time by keeping booked appointments up-to-date.

Employee: Keep appointment information up-to-date should a change occur.

Concordia University Students: Produce a system that functions as expected by the Welcome Hall Mission.

Pre-Conditions:

1. Client has phoned to update an appointment
2. Client has an existing appointment

Post Conditions:

Success end condition:

1. An appointment is updated

Failure end condition:

1. Appointment remains unchanged

Minimal Guarantee:

1. A descriptive message is displayed to the user.

Main Success Scenario:

1. Employee initiates client file search
2. System returns client file
3. Employee navigates to the appointment list section of the client file
4. System displays a list of appointments for the respective client
5. Employee opens the respective appointment
6. System displays appointment information
7. Employee makes necessary updates to the appointment and saves
8. System confirms that the changes were saved

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC3.3 Cancel Appointment

ID: UC3.3

Use Case: Cancel Appointment

Description:The employee receives a phone call from the client and starts the application. The user searches for the clients file and views their existing appointments. The employee selects the appointment to be canceled and cancels it.

Level:User Goal

Primary Actor:Employee

Supporting Actors:N/A

Stakeholders and Interests:

Welcome Hall Mission: Keep the organization organized by tracking which clients attend which events.

Client: Ensure a ban is not placed on their account if they cannot attend the event and call to cancel in advance.

Employee: Keep event booking up-to-date.

Concordia University Students: Produce a system that functions as expected by the Welcome Hall Mission.

Pre-Conditions:

1. Client has phoned to cancel an appointment
2. Client has an existing appointment

Post Conditions:

Success end condition:

1. An appointment is cancelled

Failure end condition:

1. Appointment remains unchanged
2. An error message is displayed

Minimal Guarantee:

1. A descriptive message is displayed to the user.

Main Success Scenario:

1. Employee initiates client file search
2. System returns client file
3. Employee navigates to the appointment list section of the client file
4. System displays a list of appointments for that client
5. Employee opens the respective appointment
6. System displays appointment information
7. Employee chooses to cancel the appointment
8. System confirms the cancellation of the appointment

Extensions:

1.a Search for a client using file ID number field See UC1.7.

1.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC3.4 View All Booked Appointments

**ID**: UC3.4

**Use Case:** View All Booked Appointments

**Description:** Employee wishes to view all scheduled events for a client. The employee navigates to the client file and selects the option to view scheduled appointments. The system displays a list of all appointments the user has scheduled since their files creation.

**Level:** User Goal

**Primary Actor:** Employee

**Supporting Actors:** None

**Stakeholders and Interests:**

Welcome Hall Mission: View a list of all clients attending an event.

Client: Quick and efficient check-in to the event.

Employee: Easily check-in clients to the event.

Concordia University Students: Produce a system that functions as expected by the Welcome Hall Mission.

**Pre-Conditions:**

1. Client has existing appointment(s)

**Post Conditions:**

Success end condition:

1. All appointments are displayed

Failure end condition:

1. No appointments are displayed
2. An error message is displayed

Minimal Guarantee:

1. Feedback is displayed to the user

**Main Success Scenario:**

1. Employee initiates client file search
2. System returns client file
3. Employee navigates to the appointment list section of the client file
4. System displays a list of appointments for that client

**Extensions:**

2.a Search for a client using file ID number field See UC1.7.

2.b Search for a client file using advanced search See UC1.8.

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC4.1 Get All Event Attendees

ID:UC4.1

Use Case:Get All Event Attendees

Description:Select an event and load the list of attendees registered for that event.

Level:Sea

Primary Actor:Employee

Supporting Actors:None

Stakeholders and Interests:

Employee: Be able to easily check-in all attendees to an event and ensure no overbooking.

Pre-Conditions:

1.A mobile device is available and is functioning.  
2. The application is started on the mobile device.

Post Conditions:

Success end condition:

* 1. The attendees for the event registered in all appointments are displayed on the screen in the alphabetical order sorted by the last name.

Failure end condition:

* + 1. The list of attendees has no items in it

Minimal Guarantee:

* + - 1. No data in the database is modified
      2. No change to the state of the system is made

Main Success Scenario:

1. Employee navigates to the active event list section
2. Employee selects the day for which to display the list of events
3. The system displays events registered for the selected day
4. Employee selects the event from the list

Extensions:

1.a: No wireless connection is available

1.a.1. System notifies the user that the WiFi connection is necessary to retrieve the list of events

1.a.2. Use case terminates with failure end condition

3.a: The day selected has no events registered

3.a.1. System notifies the user that no events have been scheduled for the day selected

3.a.2. Use case terminates with failure end condition

5.a: The event selected has no attendees registered

5.a.1. System notifies the user that no attendees have been registered for the event

5.a.2. Use case terminates with failure end condition

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 9 Purchased Componentsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC4.2 Appointment Fulfillment

ID:UC4.2

Use Case:Appointment Fulfillment

Description:Mark a person as having fulfilled an appointment**.** Employee finds the person’s name and indicates that he showed up. System updates the information related to this appointment.

Level:Sea

Primary Actor:Employee

Supporting Actors:Client

**Stakeholders and Interests:**

Employee: Check-in all clients who attend the event.

Client: Avoid a ban on their account by ensuring they are checked-in to an event.

Pre-Conditions:

* 1. A mobile device is available and is functioning
  2. A wireless connection with the server is established
  3. The application is started on the mobile device
  4. The current event is selected and the list of its attendees is loaded

Post Conditions:

Success end condition:

1. Appointment attendee has been marked as having his appointment fulfilled

Failure end condition:

1. The state of appointment fulfillment of all attendees remains unchanged

Minimal Guarantee:

1. The list of events registered has not changed
2. The list of appointments for all events has not changed
3. The list of attendees for all appointments across all events has not changed
4. Personal data of the attendee remains unchanged
5. No change is done to data fields outside the “Appointment Fulfillment” component boundaries

Main Success Scenario:

1. Client presents his ID or says his last name
2. Employee enters the client last name
3. As the employee enters the data, system filters the attendees list according to the prefix of the string entered
4. When the name of the client is visible in the filtered subset, Employee marks the person as “Arrived”
5. The system indicates that the command is acknowledged
6. Employee confirms the action by asking the application to commit the change of state
7. System updates the database indicating that the appointment has been fulfilled
8. System re-initializes quick-search so that the next attendee could be processed.

Extensions:

1.a: The client says his file number instead of his last name

1.a.1. Client provides his file number

1.a.2. Employee indicates the search by file number

1.a.3. Employee starts entering the file number

1.a.4. Use case continues at Step 3

4.a: The client is not on the list

4.a.1. Use case ends with a failure end condition

4.b.: Manual client search

4.b.1. Employee finds the name manually by scrolling the list down

4.b.2. Employee marks the person as “Arrived”

4.b.3. Use case continues at Step 5

6.a: The Employee has made a mistake and wishes to undo it

6.a.1. Employee un-marks person’s name as having his appointment fulfilled

6.a.2. System indicates that the command is acknowledged

6.a.3. Use case continues at Step 4

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 9 Purchased Componentsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC4.3 Print Attendee List

ID:UC4.3

Use Case:Print Attendee List

Description:Mobile devices are unable to perform their roles and the Employee acquires a physical medium of a list of attendees for an event.

Level:Sea

Primary Actor:Employee

Supporting Actors:None

Stakeholders and Interests:

Employee: Ability to manually check-in all attendees.

Client: Avoid a ban on their account by ensuring they are checked-in.

Pre-Conditions:

* 1. The work station is available and is functioning
  2. The application is running on a work station

Post Conditions:

Success end condition:

1. The list of attendees for the selected appointment is printed on paper.

Failure end condition:

1. The list of attendees is not printed from a printer.
2. A physical copy of the list is not printed and obtained.

Minimal Guarantee:

1. The list of events registered has not changed
2. The list of appointments for all events has not changed
3. The list of attendees for all appointments across all events has not changed
4. All data of the attendees remain unchanged

Main Success Scenario:

1. Employee loads the list of event attendees for an event by performing UC4.01 on the desktop version of the system
2. Employee selects to print the current list of attendees
3. System sends a message to the printer containing the list of attendees to print

Extensions:

2.a: No printer is connected

2.a.1. System prompts a message stating that no printer is connected.

2.a.2. Use case ends with a failure end condition

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### ~~UC5.1 ManageReport Templates~~

~~ID: UC5.1~~

~~Use Case:Manage Report Templates~~

~~Description:Use case describes how the Employee create, view, modify or delete a template for a particular report.~~

~~Level:User Goal Level~~

~~Primary Actor:Employee~~

~~Supporting Actors:None~~

~~Stakeholders and Interests:~~

~~Employee: Create and view custom reports.~~

~~Pre-Conditions:~~

~~Employee must be authenticated~~

~~Post Conditions:~~

~~Success end condition~~

1. ~~Template successfully created~~

~~Failure end condition:~~

1. ~~Template is not created~~

~~Minimal Guarantee~~

1. ~~Appropriate messages will be displayed.~~
2. ~~All existing templates are unaffected~~

~~Main Success Scenario:~~

1. ~~Employee navigates to the manage template section.~~
2. ~~System displays the available operations~~
3. ~~Employee submits the request to create a new template.~~
4. ~~System displays filter options (ex: event management, items management, etc)~~
5. ~~Employee selects appropriate options to create template according to user‘s requirements.~~
6. ~~System displays the options to save template.~~
7. ~~Employee saves the template.~~
8. ~~System displays appropriate message.~~

~~Extensions:~~

~~8.a Employee has not selected any options for the template~~

~~8.a.1 An error message is displayed~~

~~8.a.2 The template is not created~~

~~Special Requirements:~~

~~See Section 2 Functionalityof Supplementary Specifications document~~

~~See Section 3 Usabilityof Supplementary Specifications document~~

~~See Section 4 Reliabilityof Supplementary Specifications document~~

~~See Section 5 Performanceof Supplementary Specifications document~~

~~See Section 6 Supportabilityof Supplementary Specifications document~~

~~See Section 7 Design Constraintsof Supplementary Specifications document~~

~~See Section 10 Interfacesof Supplementary Specifications document~~

\*This feature has been scoped out by the client.

### UC5.2 Generate Operational Reports

ID:UC5.2

Use Case: Generate Operational Reports

Description:Employee can generate Operational reports by giving the time span and selecting the export option.

Level:User Goal Level

Primary Actor:Employee

Supporting Actors:None

Stakeholders and Interests:

Employee: Generate predefined reports.

Pre-Conditions:

Employee must be authenticated

Post Conditions:

Success end condition

1. Operational reports successfully generated and exported

Failure end condition

1. Report generation and exporting fails.

Minimal Guarantee

1. Appropriate messages will be displayed.

Main Success Scenario:

1. The use case begins when employee triggers manage operational report event.
2. Employee submits the request to generate a new report from pre-generated templates or by creating new report by giving the time span.
3. System displays filter options (ex: event management, items management, etc)
4. Employee selects appropriate options
5. System generates the report and display it to the employee.
6. Employee chooses to export the generated report.
7. System exports the report.

Extensions:

5.a No filter options were displayed

5.a.1 The report is not generated

5.a.2 An error message is displayed to the client

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

### UC5.3 Generate Statistical Reports

ID:UC5.3

Use Case: Generate Statistical Reports

Description:Employee wishes to generate a statistical report in regards to client attendance to events byy specifying a date range.

Level:User Goal Level

Primary Actor:Employee

Supporting Actors:None

Stakeholders and Interests:

Employee: Generate predefined reports.

Pre-Conditions:

1. Employee must be authenticated

Post Conditions:

Success end condition

* 1. Operational reports successfully generated.
  2. Excel version downloaded to the users’ computer.

Failure end condition

1. Eror messagwe displayed to user indicating that the report could not be generated.

Minimal Guarantee

1. Appropriate messages will be displayed.

Main Success Scenario:

1. The employee triggers manage statistical report event.

2. Employee submits the request to generate a new report for a given time span.

3. System displays filter options (ex: user, area of Montreal, country of origin, etc)

4. Employee selects appropriate options

5. System generates the report and display it to the employee.

6. Employee chooses to export the generated report.

7. System exports the report.

Extensions:

5.a No filter options were supplied

5.a.1 The report is not generated

5.a.2 An error message is displayed to the client

Special Requirements:

See Section 2 Functionalityof Supplementary Specifications document

See Section 3 Usabilityof Supplementary Specifications document

See Section 4 Reliabilityof Supplementary Specifications document

See Section 5 Performanceof Supplementary Specifications document

See Section 6 Supportabilityof Supplementary Specifications document

See Section 7 Design Constraintsof Supplementary Specifications document

See Section 10 Interfacesof Supplementary Specifications document

# Logical View

## Operational Contracts

### CO1.1 Register a Client

**Contract:**CO1.1 Register a Client

**Operation:**ClientController.actionCreate(Client c, Residence r)

**Cross References:** UC1.1

**Preconditions:** A client registration is underway.

**Postconditions:**

A *clientFile* instance was created (instance creation).

*clientFile.id* was set to ID (attribute modification).

*clientFile.f\_name* was set to First Name (attribute modification).

*clientFile.l\_name* was set to Last Name (attribute modification).

*clientFile.medicare* was set to Medicare (attribute modification).

*clientFile.mother\_tongue* was set to Mother Tongue (attribute modification).

*clientFile.language\_of\_comm\_id* was set to Spoken (attribute modification).

*clientFile.other\_languages* was set to Other Languages (attribute modification).

*clientFile,residence\_id* was set to Residence (attribute modification).

*clientFile.country\_of\_origin\_id* was set to Origin (attribute modification).

*clientFile.citizen\_status\_id* was set to Citizenship (attribute modification).

*clientFile.referal\_id* was set to Referral (attribute modification).

*clientFile.regwastration\_date* was set to Regwastration Date (attribute modification).

*clientFile.phone\_number* was set to Phone # (attribute modification).

*clientFile.email* was set to Email (attribute modification).

*clientFile.gender* was set to Gender (attribute modification).

*clientFile.last\_date\_vwasited* was set to Last Vwasit (attribute modification).

*clientFile.last\_date\_modified* was set to Modified On (attribute modification).

*clientFile.marital\_status\_id* was set to Marital Status (attribute modification).

*clientFile.work\_status\_id* was set to Work Status (attribute modification).

*clientFile.initials\_created* was set to Created By (attribute modification).

*clientFile.initials\_last\_modified* was set to Updated By (attribute modification).

### CO1.2 Register a Dependent

**Contract:**CO1.2 Register a Dependent

**Operation:** DependantController.actionCreate(Dependent d)

**Cross References:** UC1.9

**Preconditions:** A client registration is underway.

**Postconditions:**

A *dependent* instance was created (instance creation).

*Dependent*.id was set to ID (attribute modification).

*Dependent*.f\_name was set to First Name (attribute modification).

*Dependent*.l\_name was set to Last Name (attribute modification).

*Dependent*.medicare was set to Medicare (attribute modification).

*Dependent*.gender was set to Gender (attribute modification).

*Dependent*.relationship\_id was set to Relation (attribute modification).

*Dependent*.citizen\_status\_id was set to Citizenship (attribute modification).

*Dependent*.work\_status\_id was set to Work Status (attribute modification).

*Dependent*.initials\_last\_modified was set to Updated By (attribute modification).

*Dependent* was associated with *clientFile*(association formed)*.*

### CO1.3 Deregister a Dependent

**Contract:**CO1.3 Register a Dependent

**Operation:** DependantController.actionDelete(Dependent d)

**Cross References:** UC1.10

**Preconditions:** A client deregistration is underway.

**Postconditions:**

A *dependent* instance was deleted (instance deletion).

*Dependent* was disassociated from *clientFile* (association broken)*.*

### CO1.4 Create Flag

**Contract:**CO1.4Create Flag

**Operation:** FlagController.actionCreate(Flag f)

**Cross References:** UC1.5

**Preconditions:** Client file modification is underway.

**Postconditions:**

A *flag* instance was created (instance creation).

*Flag.id* was set to *ID* (attribute modification).

*Flag.name* was set to *Name* (attribute modification).

*Flag.description* was set to *Description* (attribute modification).

*Flag.level* was set to *Level of Importance* (attribute modification).

*Flag.is\_active* was set to *Is Active* (attribute modification).

*Flag* was associated with *clientFile* (association formed)*.*

### CO1.5 Remove Flag

**Contract:**CO1.5Remove Flag

**Operation:** FlagController.actionDelete(Flag f)

**Cross References:** UC1.12

**Preconditions:** Client file modification is underway.

**Postconditions:**

A *flag* instance was deleted (instance deletion).

*Flag* was disassociated with *clientFile* (association broken)*.*

### CO1.6 Add Note

**Contract:**CO1.6Add Note

**Operation:** NoteController.actionCreate(Note n)

**Cross References:** UC1.4

**Preconditions:** Client file modification is underway.

**Postconditions:**

A *note* instance was created (instance creation).

*Note.id* was set to ID (attribute modification).

*Note.client\_id* was set to Client ID (attribute modification).

*Note.initials* was set to By (attribute modification).

*Note.date\_modified* was set to Date Modified (attribute modification).

*Note.note\_text* was set to Note Text (attribute modification).

*Note* was associated with *clientFile* (association formed)*.*

### CO2.1 Create Event Template

**Contract:**CO2.1 Create Event Template

**Operation:** EventController.actionCreate(EventTemplate e)

**Cross References:** UC2.1

**Preconditions:** System configuration is underway.

**Postconditions:**

A *eventTemplate* instance was created (instance creation).

*eventTemplate.id* was set to ID (attribute modification).

*eventTemplate.name* was set to Name (attribute modification).

*eventTemplate.repeatable* was set to Repeatable (attribute modification).

*eventTemplate.weekday* was set to Weekday (attribute modification).

*eventTemplate.repeat\_times* was set to Repeat Times (attribute modification).

*eventTemplate*.*appointmentsTakenBy* was set to Appointments Taken By (attribute modification).

*eventTemplate.appointmentNeeded* was set to Appointment Needed (attribute modification).

*eventTemplate.timeslot\_duration* was set to Timeslot Duration (attribute modification).

*eventTemplate.start\_time* was set to Start Time (attribute modification).

*eventTemplate.end\_time* was set to End Time (attribute modification).

*eventTemplate.day\_duration* was set to Day Duration (attribute modification).

### CO2.2 Delete Event Template

**Contract:**CO2.2 Create Event Template

**Operation:** EventController.actionDelete(EventTemplate e)

**Cross References:** UC2.3

**Preconditions:** System configuration is underway.

**Postconditions:**

A *eventTemplate* instance was deleted (instance deletion).

*eventTemplate*was disassociated from *eventOccurrences*(association broken)*.*

### CO2.3 Create Event Occurrence

**Contract:**CO2.3 Create Event Occurrence

**Operation:** Event\_OccurrenceController.actionCreate(EventOccurrence e)

**Cross References:** UC2.6

**Preconditions:** Event planning is underway.

**Postconditions:**

A *eventOccurrence* instance was created (instance creation).

*eventOccurrence.id* was set to ID (attribute modification).

*eventOccurrence.event\_id* was set to Event (attribute modification).

*eventOccurrence.start\_date* was set to Start Date (attribute modification).

*eventOccurrence.end\_date* was set to End Date (attribute modification).

*eventOccurrence.repeatable* was set to Repeatable (attribute modification).

*eventOccurrence.weekday* was set to Weekday (attribute modification).

*eventOccurrence.repeat\_times* was set to Repeat Times (attribute modification).

*eventOccurrence.appointmentsTakenBy* was set to Appointments Taken By (attribute modification).

*eventOccurrence.appointmentNeeded* was set to Appointment Needed (attribute modification).

*eventOccurrence.timeslot\_duration*was set to Timeslot Duration (attribute modification).

*eventOccurrence.start\_time* was set to Start Time (attribute modification).

*eventOccurrence.end\_time* was set to End Time (attribute modification).

*eventOccurrence.isOpen* was set to Is Open (attribute modification).

*eventOccurrence.created\_on* was set to Created On (attribute modification).

*eventOccurrence.created\_by* was set to Created By (attribute modification).

*eventOccurrence* was associated with *eventTemplate*(association formed)*.*

### CO2.4 Delete Event Occurrence

**Contract:**CO2.3Cancel Event Occurrence

**Operation:** Event\_OccurrenceController.actionDelete(EventOccurrence e)

**Cross References:** UC2.9

**Preconditions:** Event planning is underway.

**Postconditions:**

A *eventOccurrence* instance was deleted (instance deletion).

*eventOccurrence* was disassociated with *eventTemplate*(association broken)*.*

### CO3.1 Book an Appointment

**Contract:**CO3.1Book an Appointment

**Operation:** AppointmentController.actionCreate(Appointment a)

**Cross References:** UC3.1

**Preconditions:** Event booking is underway.

**Postconditions:**

A *appointment* instance was created (instance creation).

*Appointment.id* was set to ID(attribute modification).

*Appointment.client\_id* was set to Client(attribute modification).

*Appointment.event\_occurrence\_timeslot\_id* was set to Event Occurrence Timeslot(attribute modification).

*Appointment.attended* was set to Attended(attribute modification).

*Appointment* was associated with *eventOccurrence*(association formed)*.*

### CO3.2 Cancel an Appointment

**Contract:**CO3.2Cancel an Appointment

**Operation:** AppointmentController.actionDelete(Appointment a)

**Cross References:** UC3.3

**Preconditions:** Appointment cancellation is underway.

**Postconditions:**

A *appointment* instance was deleted (instance deletion).

*Appointment.id* was set to ID(attribute modification).

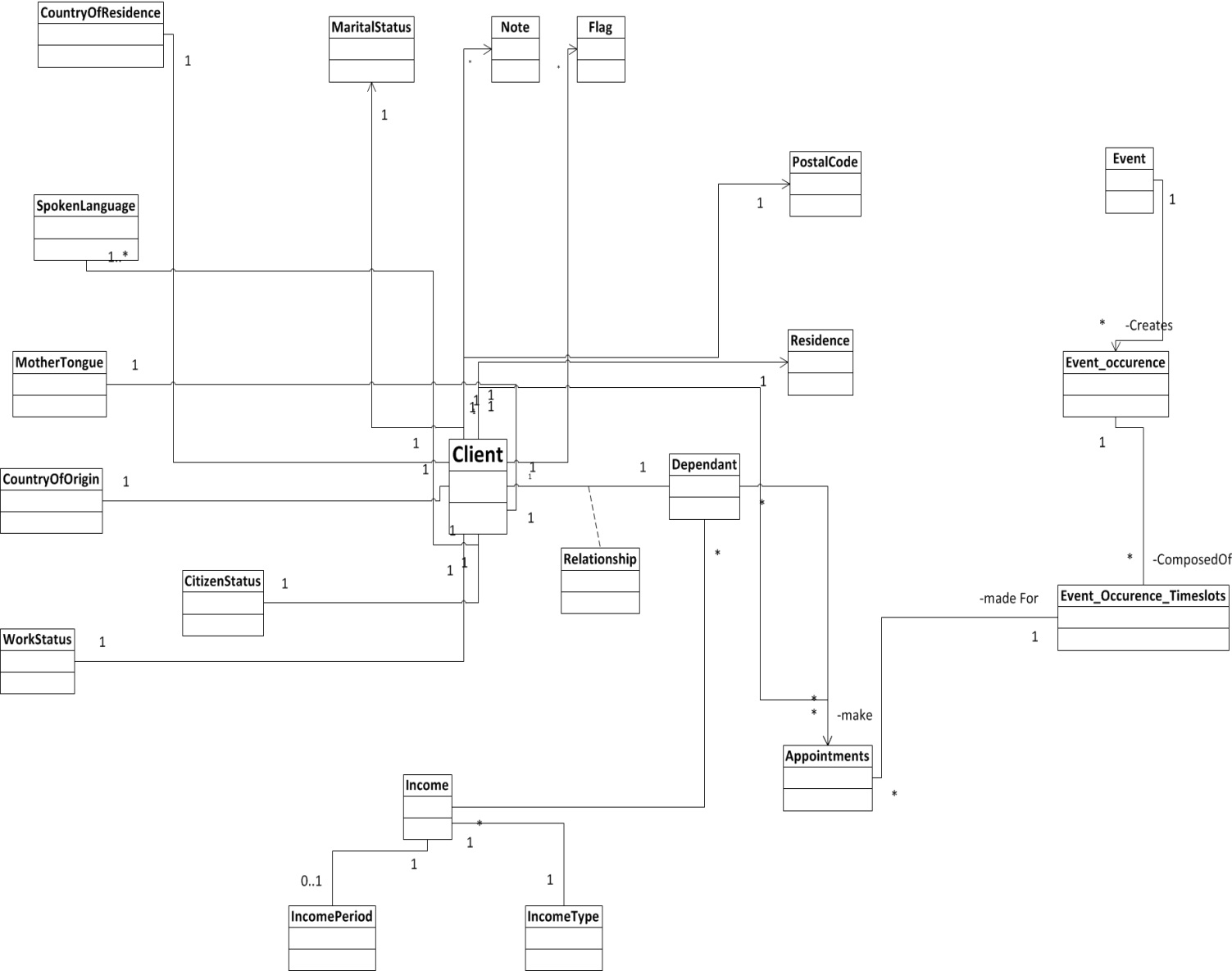
*Appointment.client\_id* was set to Client(attribute modification).

*Appointment.event\_occurrence\_timeslot\_id* was set to Event Occurrence Timeslot(attribute modification).

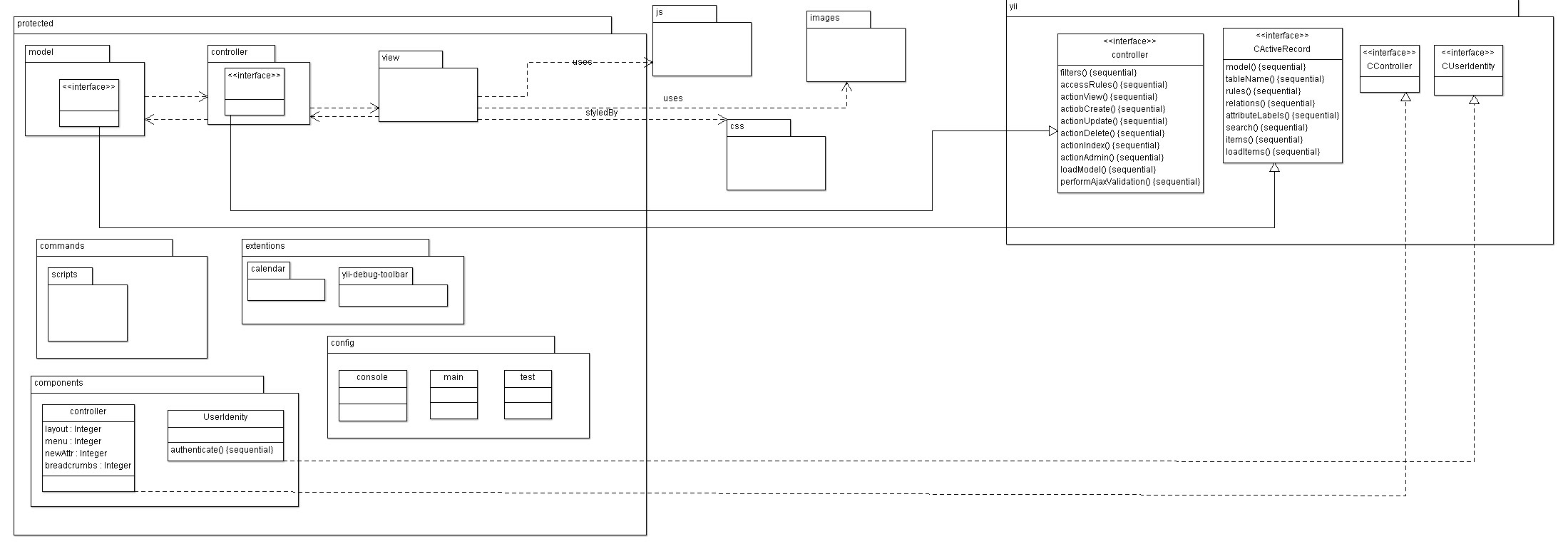
*Appointment.attended* was set to Attended(attribute modification).

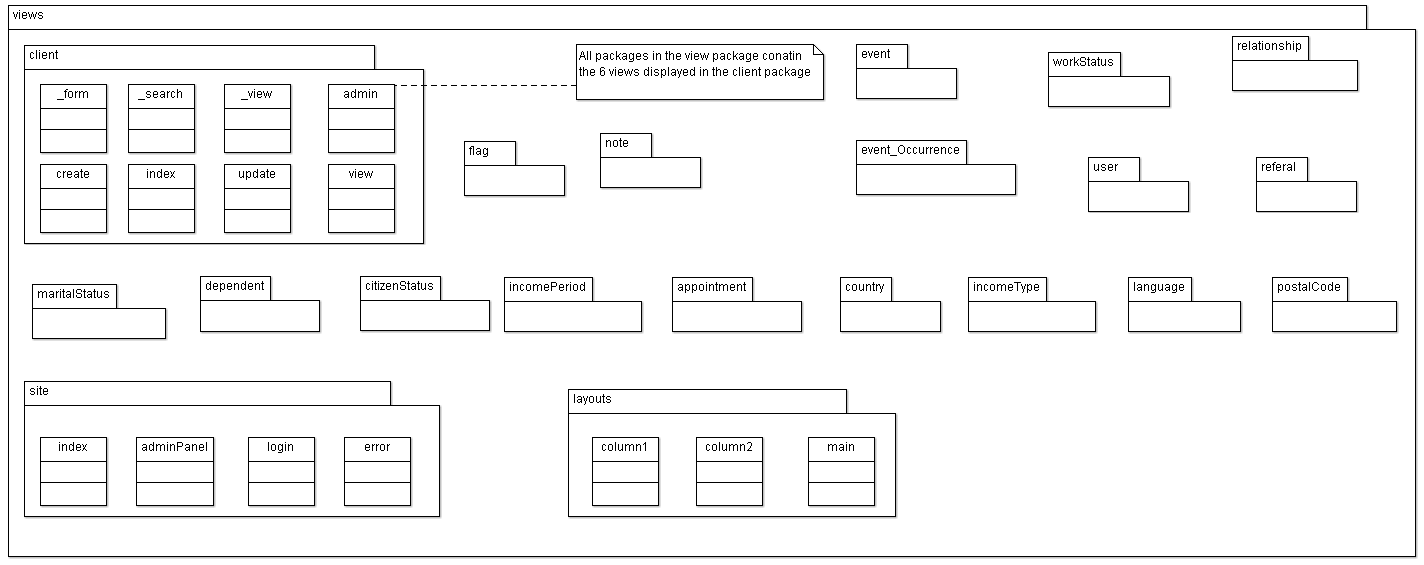
*Appointment* was disassociated with *eventOccurrence*(association broken)*.*

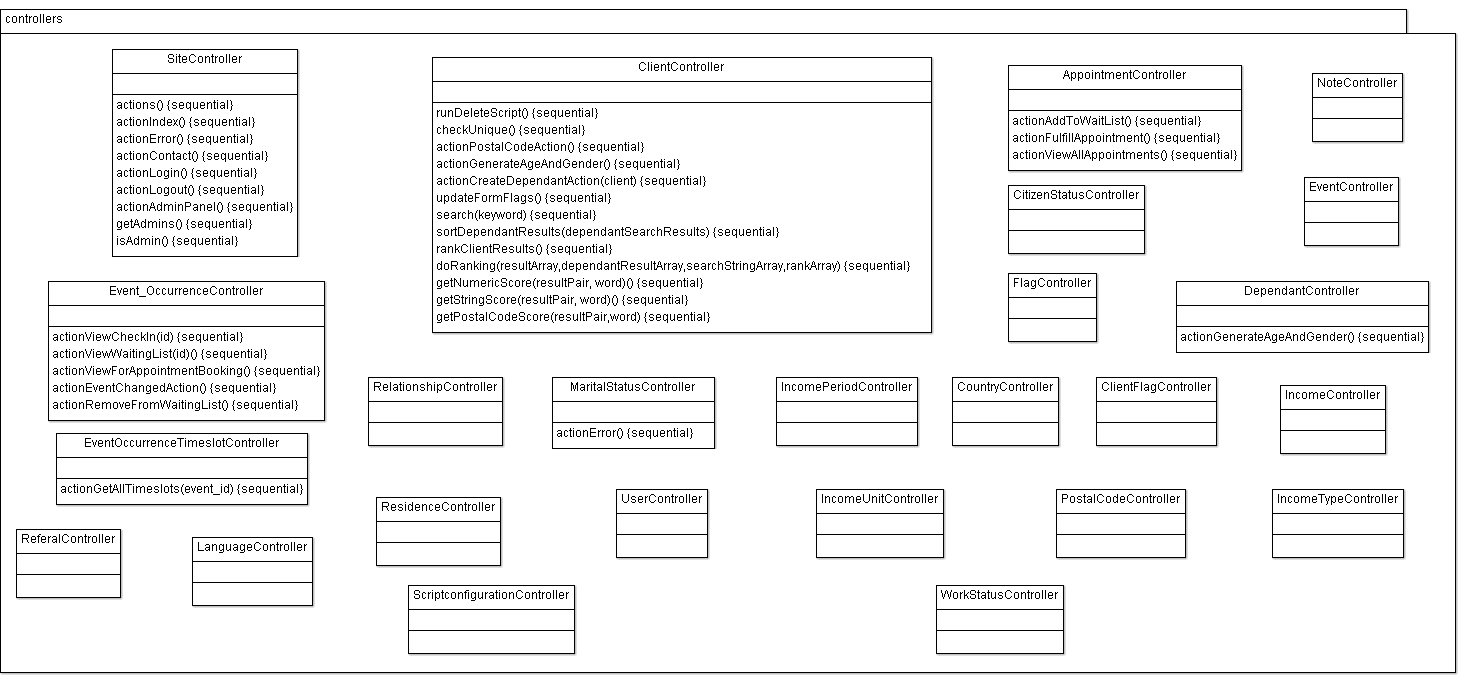
## Domain Model

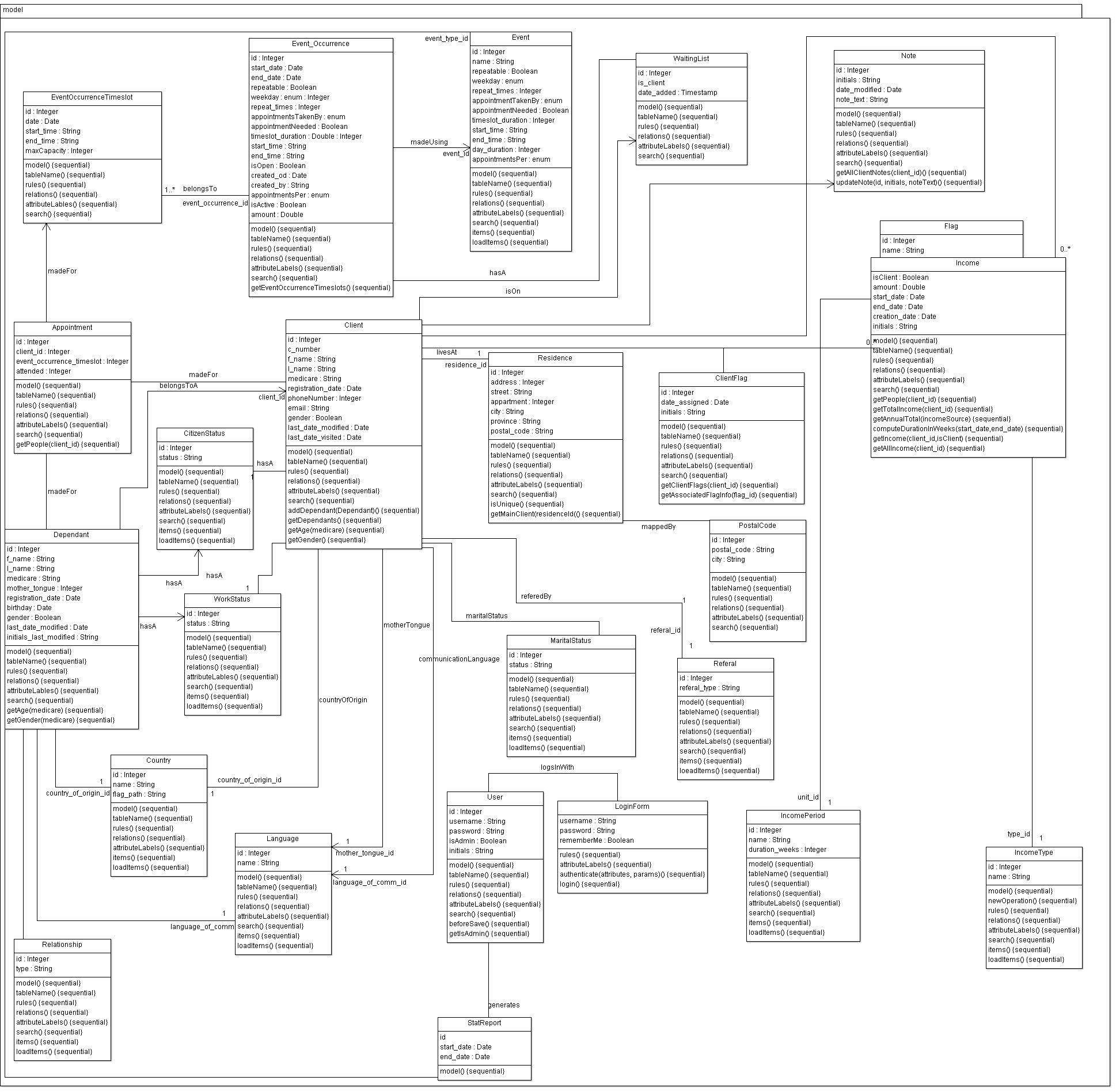


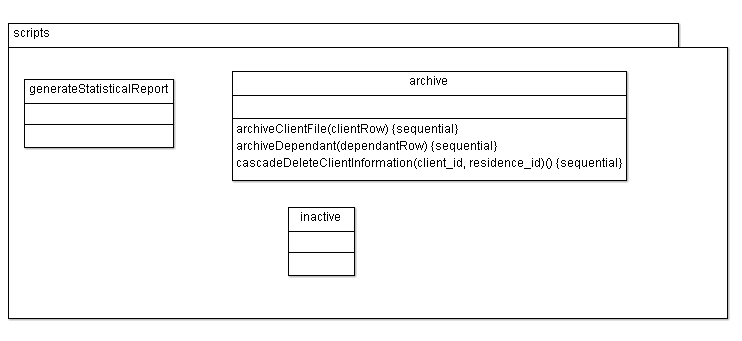
## Class Diagram





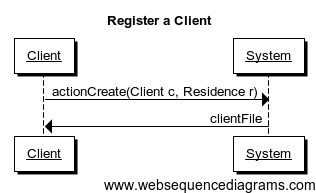




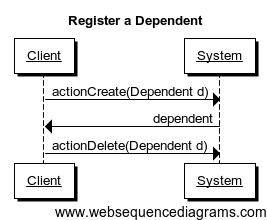


## System Sequence Diagrams

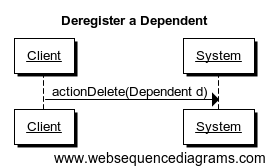
### SSD1.1 Register a Client



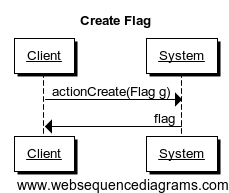
### SSD1.2 Register a Dependent



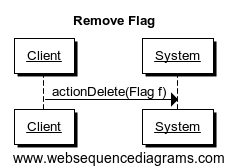
### SSD1.3 Deregister a Dependent



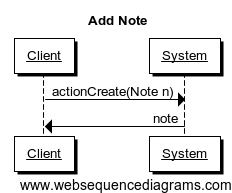
### SSD1.4 Create Flag



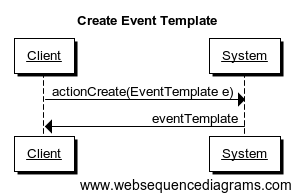
### SSD1.5 Remove Flag



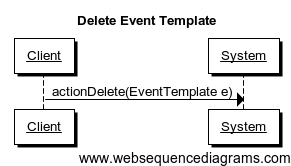
### SSD1.6 Add Note



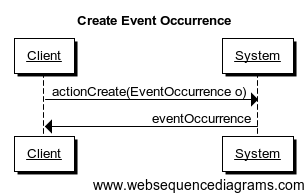
### SSD2.1 Create Event



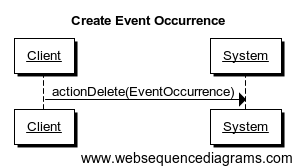
### SSD2.2 Delete Event



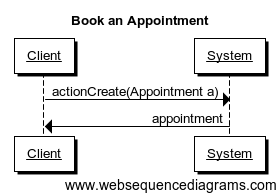
### SSD2.3 Create Event Occurrence



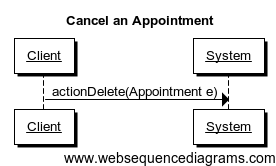
### SSD2.4 Delete Event Occurrence



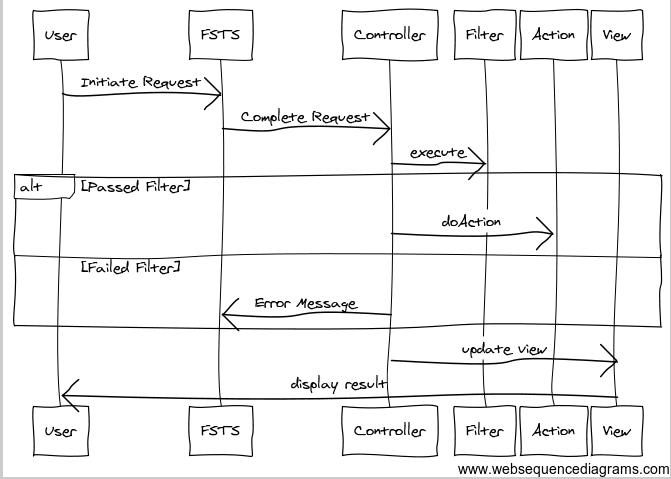
### SSD3.1 Book Appointment



### SSD3.2 Cancel Appointment

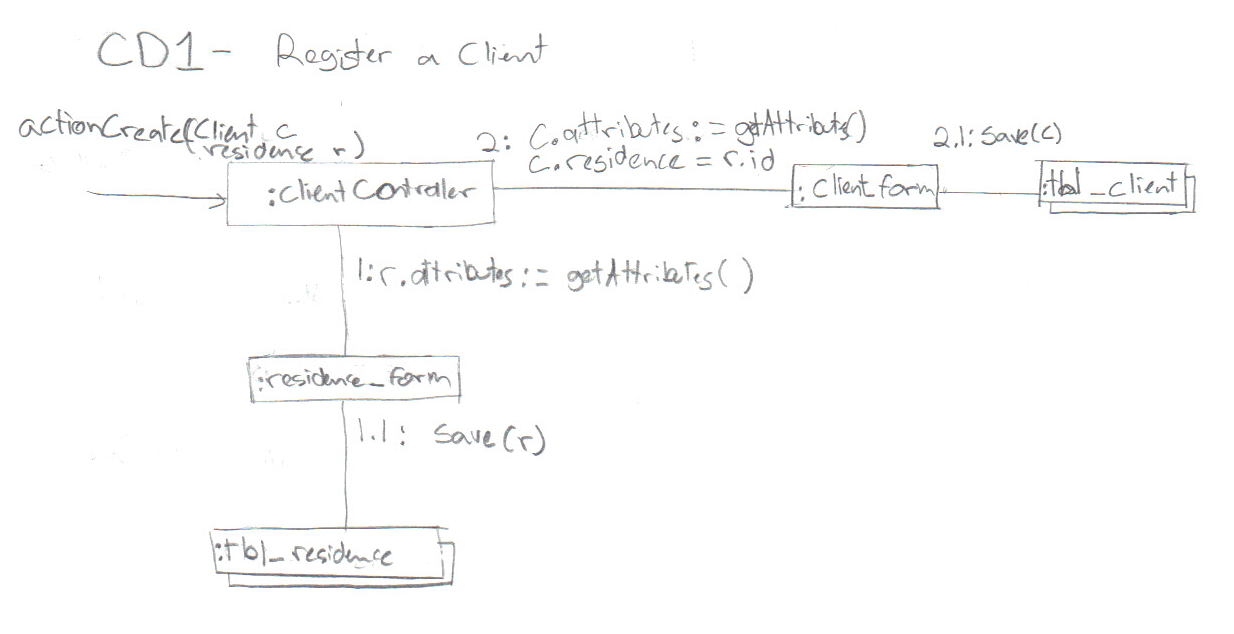


## Sequence Diagram

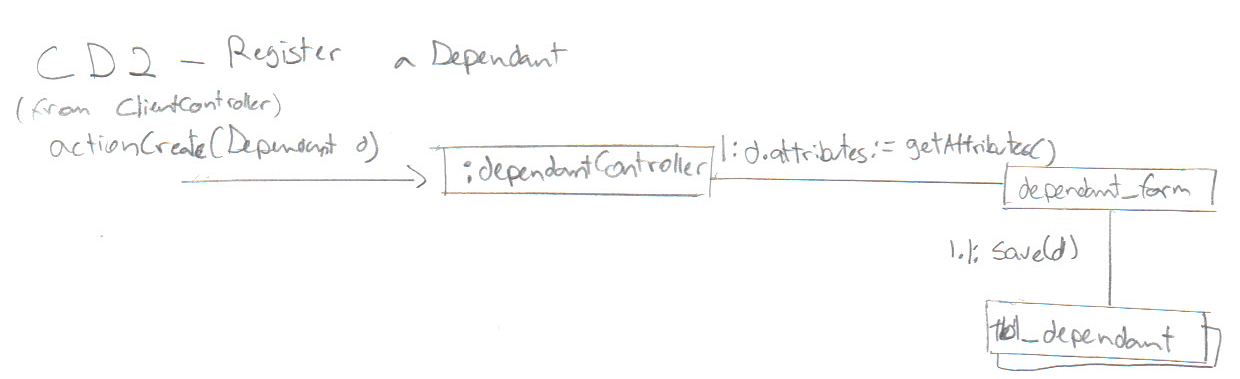


## Communication Diagrams

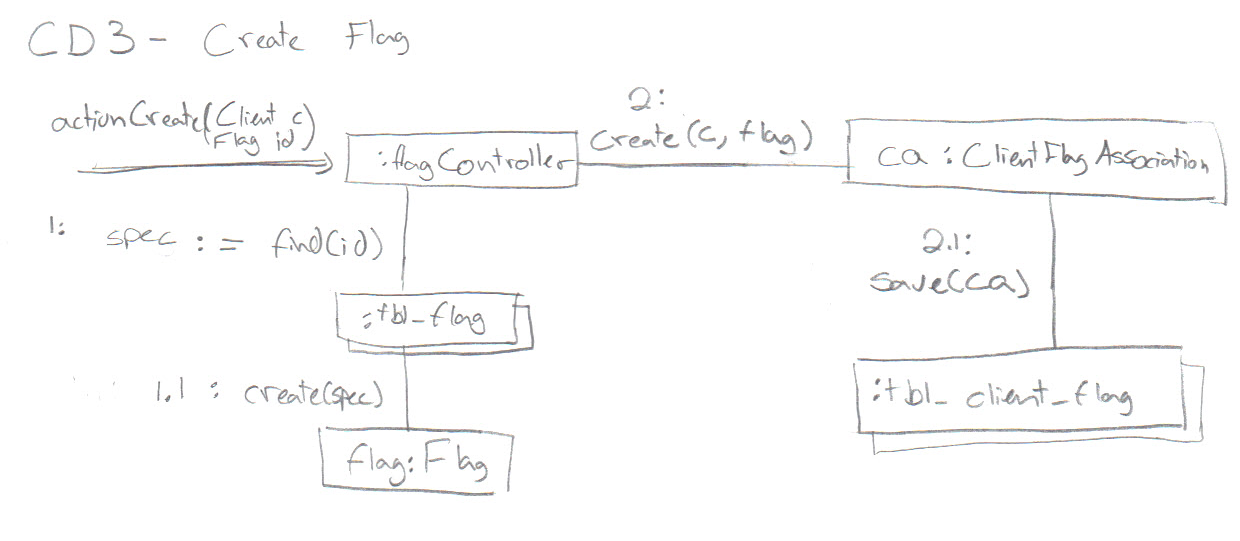
### CD1.1 Register a Client



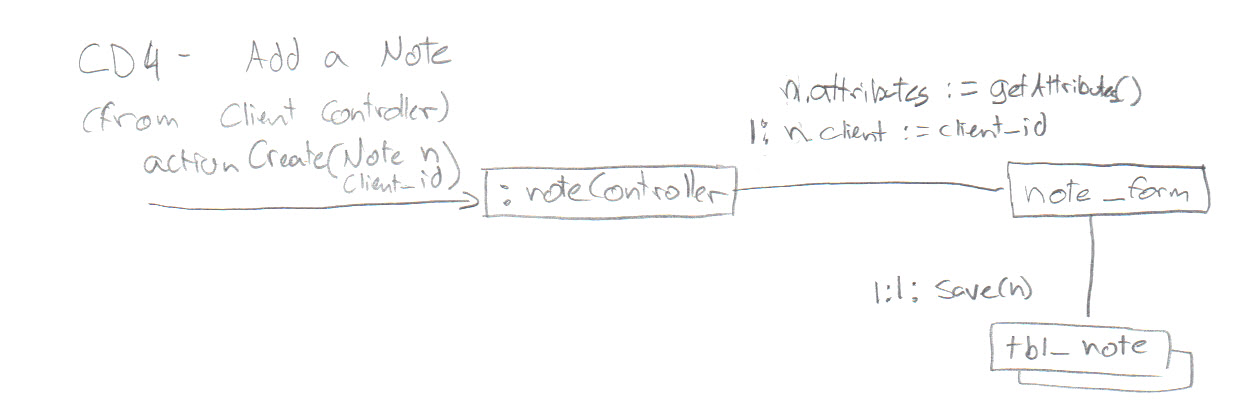
### CD1.2 Register a Dependent



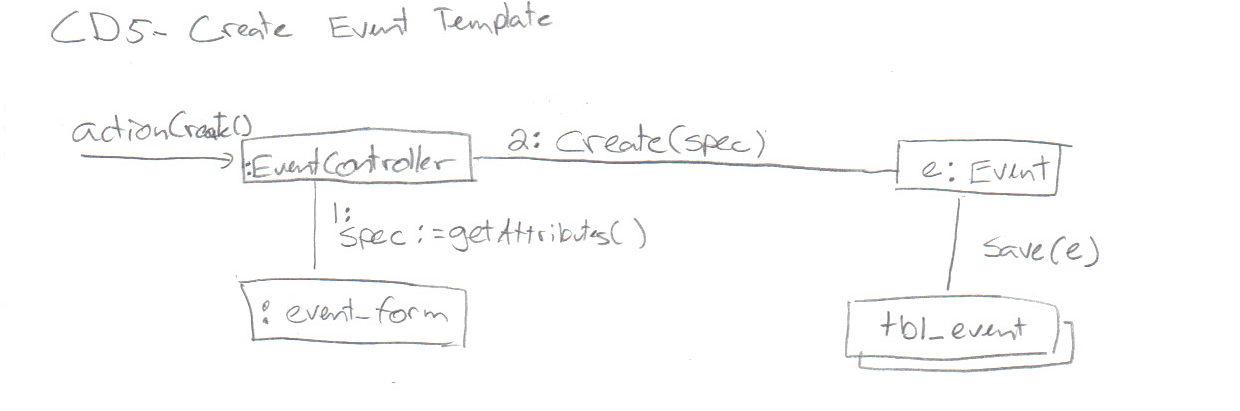
### CD1.4 Create a Flag



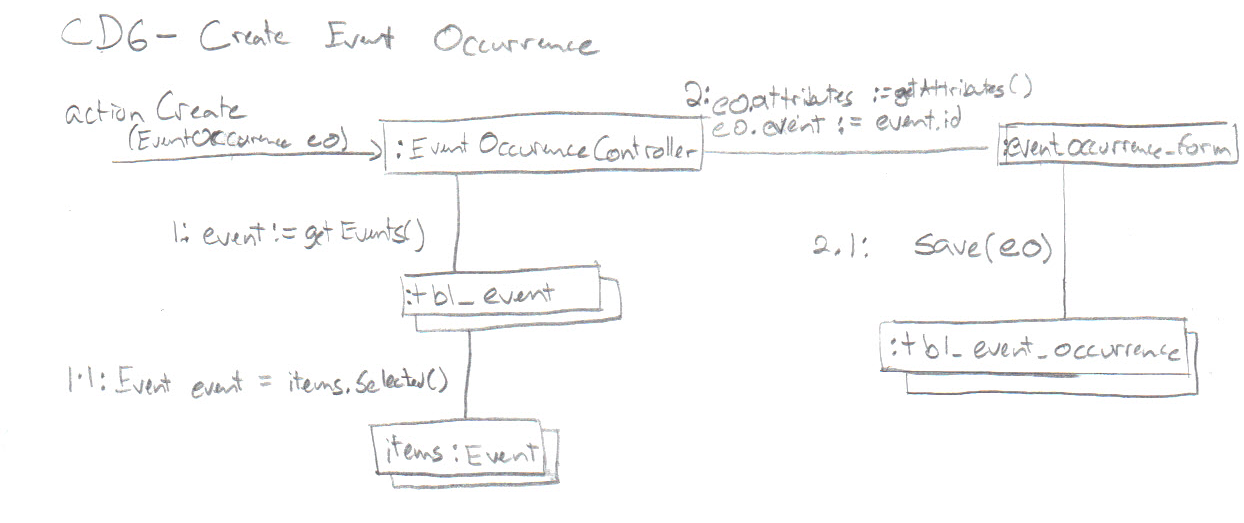
### CD1.6 Add a Note



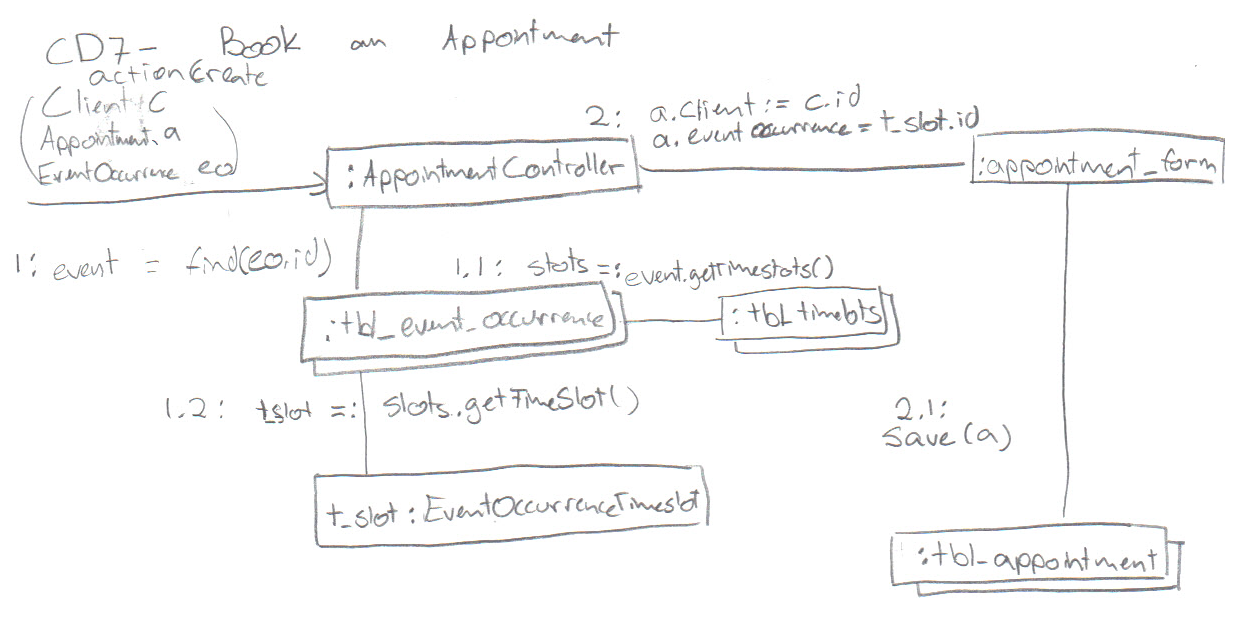
### CD2.1 Create Event Template



### CD2.3 Create Event Occurrence

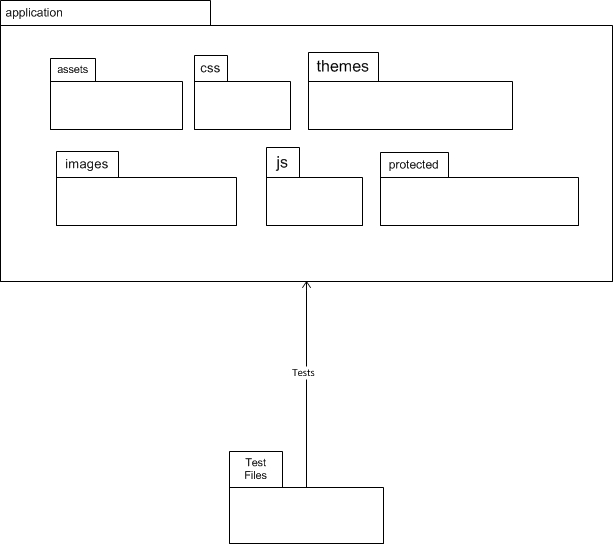


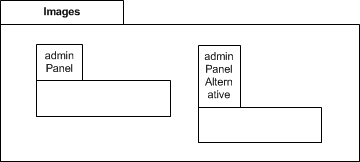
### CD3.1 Book an Appointment

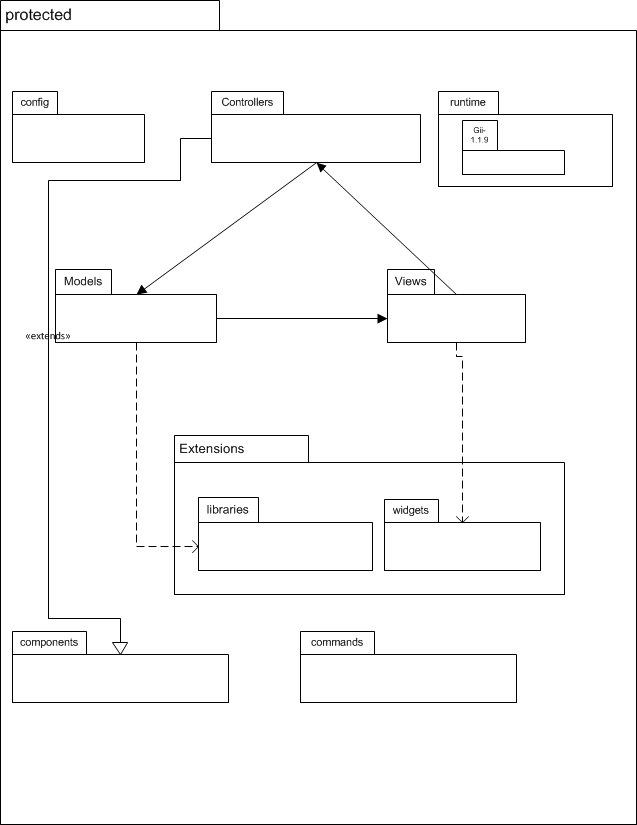


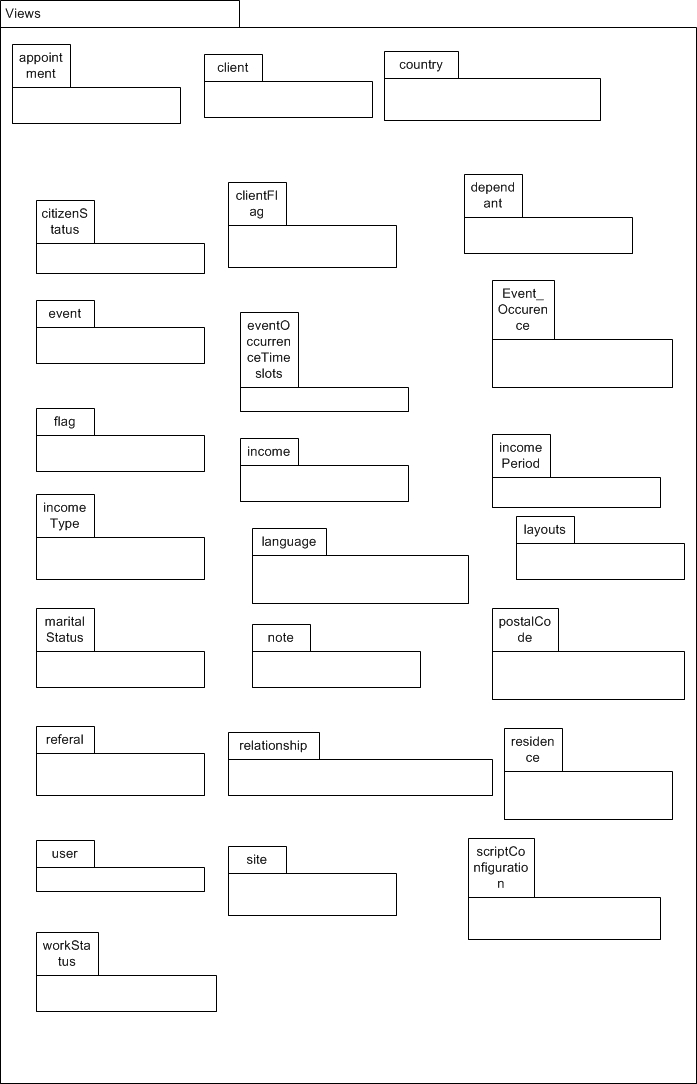
# Development View

## Package Diagram

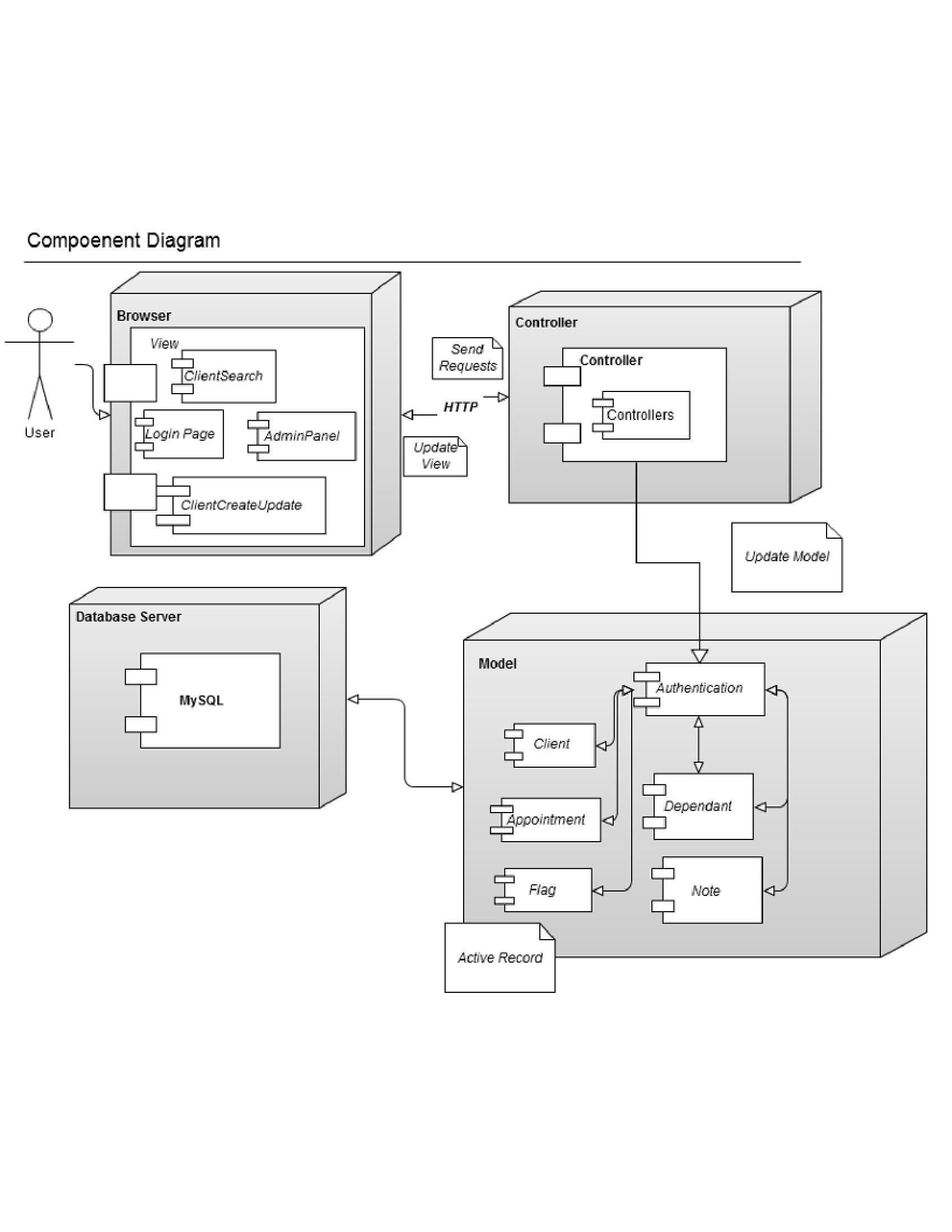








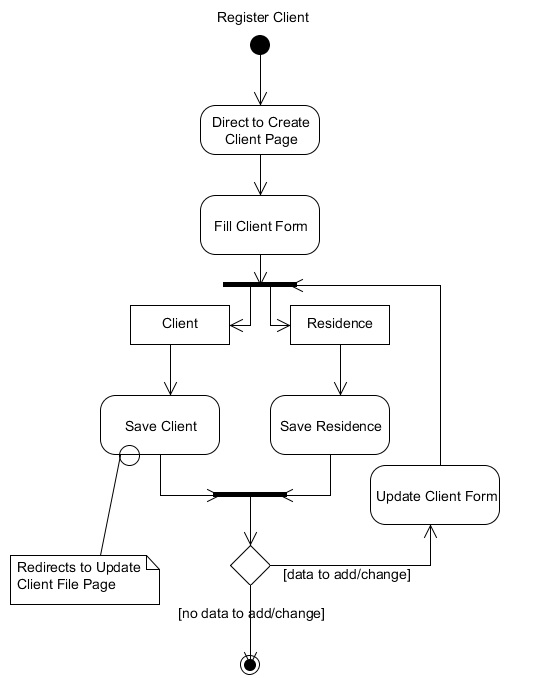
## Component Diagram



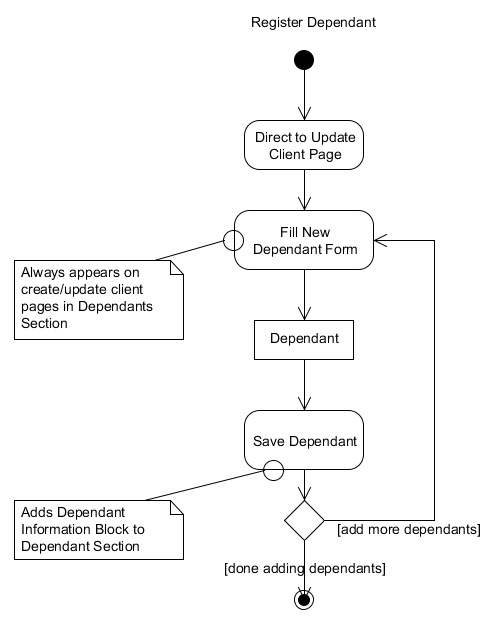
# Process View

## Activity Diagrams

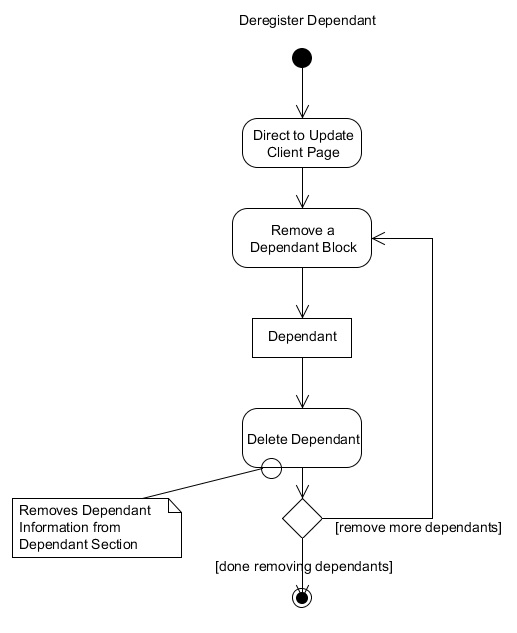
### AD1.1 Register a Client



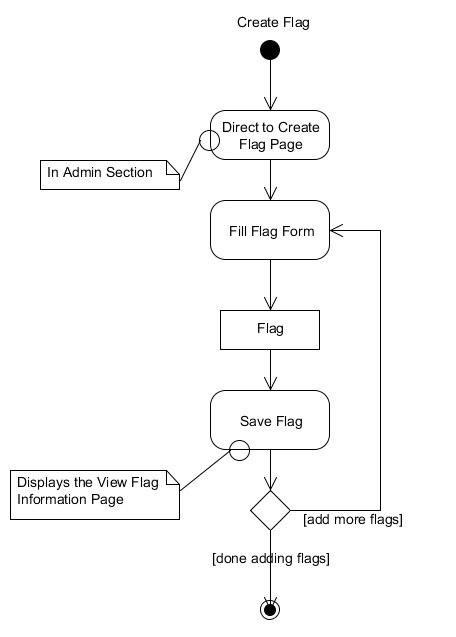
### AD1.2 Register a Dependent



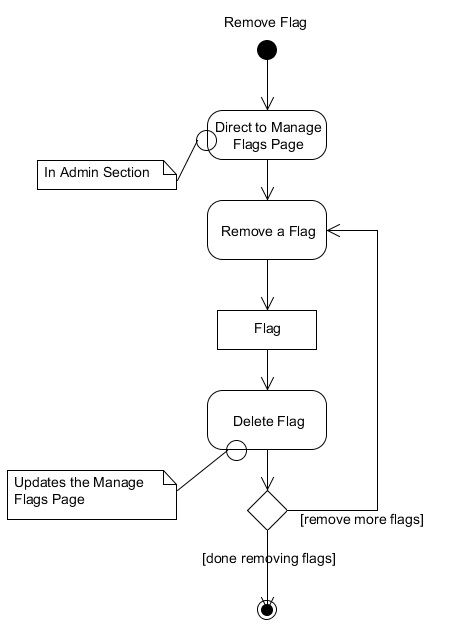
### AD1.3 Deregister a Dependent



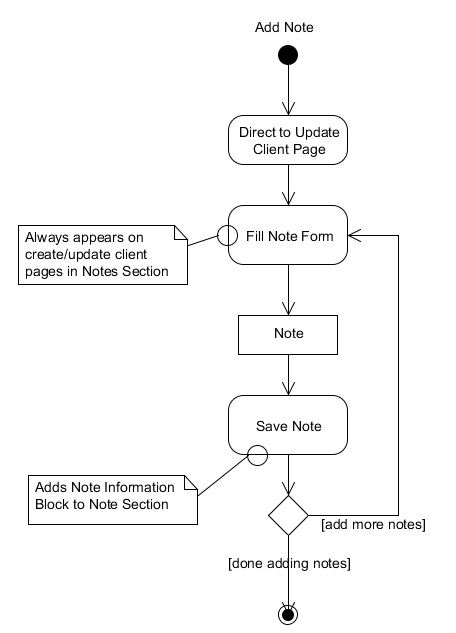
### AD1.4 Create Flag



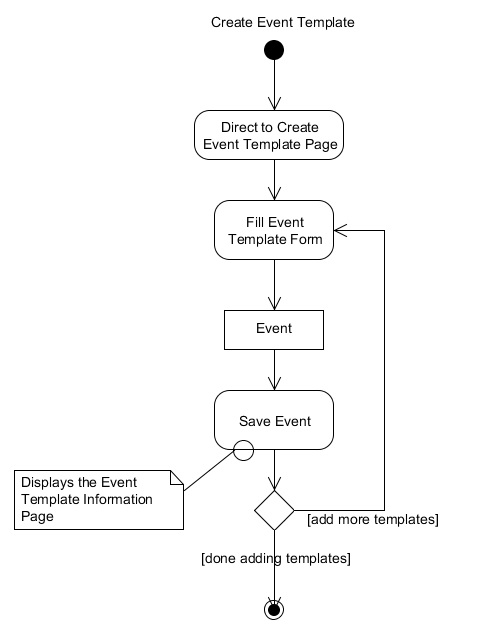
### AD1.5 Remove a Flag



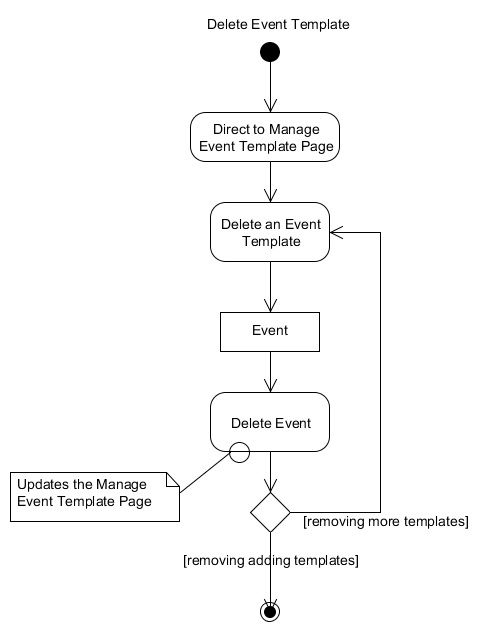
### AD1.6 Add a Note



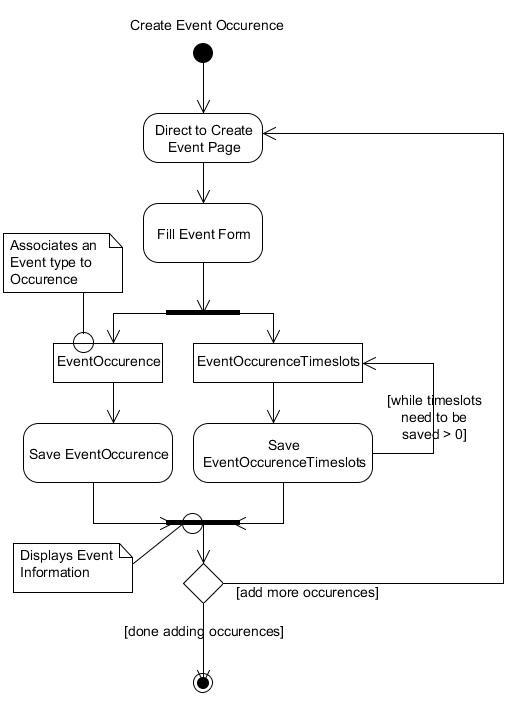
### AD2.1 Create Event Template



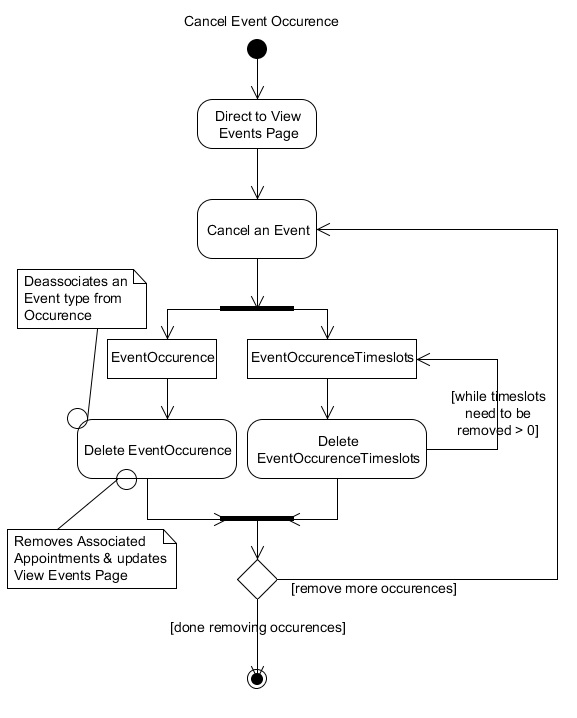
### AD2.2 Delete Event Template



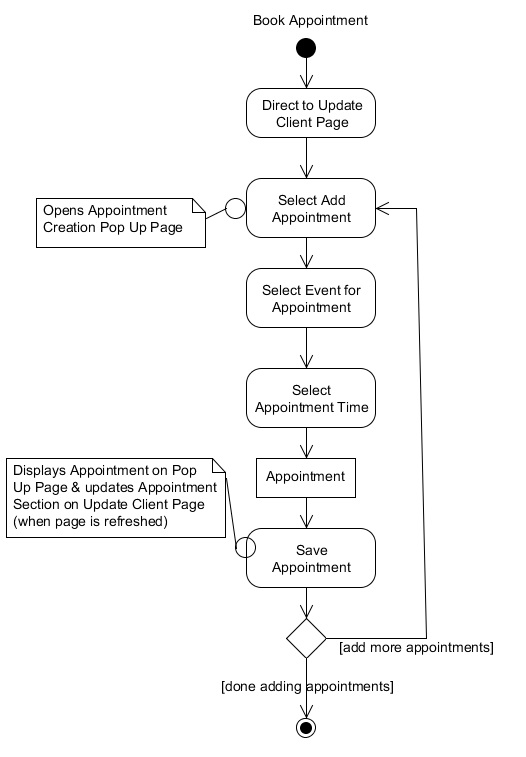
### AD2.3 Create Event Occurrence



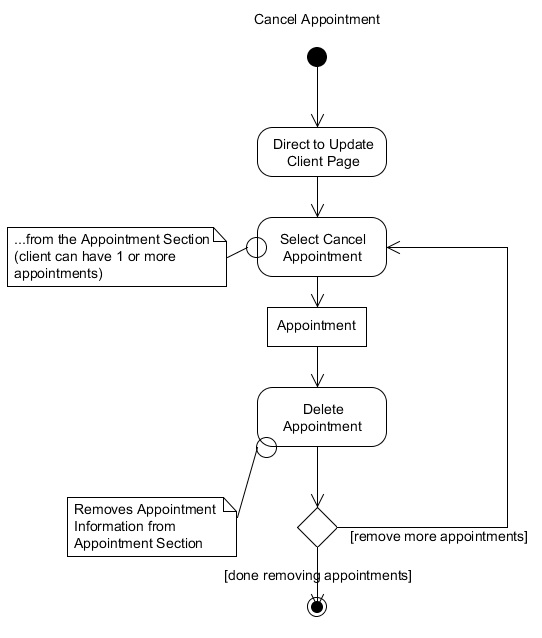
### AD2.4 Delete Event Occurrence



### AD3.1 Book an Appointment

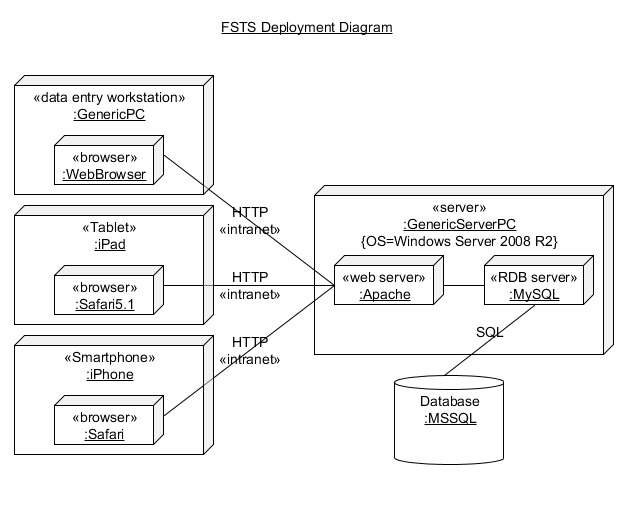


### AD3.2 Cancel an Appointment



# Physical View

## Deployment Diagram

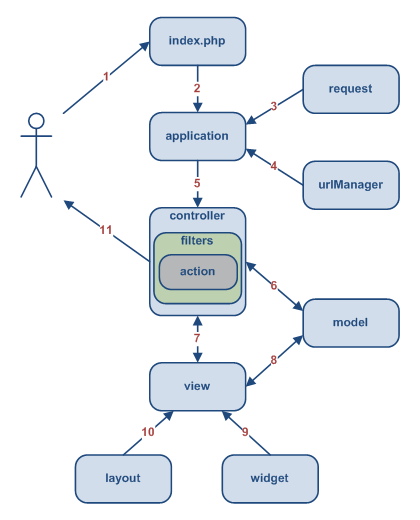


# Yii Framework Diagrams

## Yii Static Structure



## Workflow Graph



# ****Size & Performance****

The major dimensioning characteristics of the software that have an impact on the architecture can all be related to the Yii framework. The Yii framework by default contains a lot of pre made files and folders which give the developer a location to place his/her own code. So although you can create as many models, controllers and views, the architecture will not be compromised because the package structure is all determined for the user.

The Yii framework is also known for its fast performance. It only loads the features that are needed and has powerful caching support. For more information on how Yii helps with performance reference the manual: <http://www.yiiframework.com/performance/>

Please see PartVIII: Follow Up Questions in the Supplementary Specification document in regards to physical size.

Please see Part5: Performance in the Supplementary Specification document in regards to performance size.

# ****Quality****

F.S.T.S. was developed with quality in mind. The Yii framework helps out by promoting and ensuring a lot of the charactersitics that help improve the quality of our architecture.

The system can be ported easily from one environment to another, simply by changing one main configuration file. In this configuration file, you can change the type of database that is being used, and since all the database calls are abstracted there is no hardcoded mysql/mssql statements that would need to be changed.

The system is also scalable since we can add components and plug them in by simply following the architectural style provided by Yii.

Since this application will be working on an intranet, and there are no immediate or longterm plans on changing this, the dangers of outside hackers or malicious users were not explicitly addressed (although Yii does provide protection). Qualities like security and privacy were left at a default because this application would not be connected to the outside world.

Please see Part 4:Reliability in the Supplementary Specification document.

# ****Appendix A References****

[1] W3C. (2012, January 15). *HTML & CSS Standards*[Online]. Available:http://www.w3.org/standards/webdesign/htmlcss#specifications

**[2] The jQuery Project. (2012, January 15). jQuery [Online]. Available:** http://jquery.com/

[3] Yii Software LLC. (2012, January 15) *Yii PHP Framework* [Online]. Available: *http://www.yiiframework.com/*

[4] The PHP Group. (2012, January 15) *PHP* [Online]. Available: <http://www.php.net/>

[5] C. Larman. *Applying UML and Patterns*, 3rd edition. Prentice Hall. 2005.

[6] Oracle Corporation. (2012, January 12) *Netbeans IDE 7.1* [Online] Available: http://netbeans.org/index.html.

[7] Yii Software LLC. (2012, January 12) *Yii Framework; Unit Testing.*[Online] Available: http://www.yiiframework.com/doc/guide/1.1/en/test.unit

**[8] D. Rethans.** (2012, January 12) **XDebug Extension For PHP [Online]** Available: http://xdebug.org/

**[9] IEEE.** (2012, January 12) *Test Plan Outline (IEEE 829 Format*) [Online] Available: http://www.gerrardconsulting.com/tkb/guidelines/ieee829/main.html#9

[10] M. Pichler. (2012,January 12) *PHP Depend; Documentation* [Online] Available : http://pdepend.org/documentation/getting-started.html

[11] O. Ormandjieva. (2012, February 24) *Lecture14;Integration Testing* [Online]: Available: http://users.encs.concordia.ca/~s345\_4/

**[12] Selenium Project. (2012, February 23). Selenium Documentation [Online] Available :** http://seleniumhq.org/docs/

[13] CSS-Tricks. (2012, March 23). *A web design community* [Online]. Available: http://css-tricks.com/label-placement-on-forms/