Project: A.I.M. - Automated Interview Manager

CST 326, Software Engineering – Winter 2014

5 PROGRAMMERS OF TOMORROW

Team Lead: Chase Marcum
QA Lead: John Price
Architect: Jordan Ringo
Requirements Lead: Patrick Jarofski
SC/Documentation Specialist: Josh Hartwell

Table of Contents

Table of Figures

Overview

Project: A.I.M. - Automated Interview Manager Executive Summary

5 Programmers Of Tomorrow (5 P.O.T.) with the Automated Interview Manager (A.I.M.) project, aims to accomplish an automated web based kiosk system that would replace AES's current manual job applicant screening process and incorporate a new centralized hiring department that AES has created. This A.I.M system will remove the significant amount of time AES retail managers use to screen unqualified applicants and reduce the frequency that unqualified applicants are hired and then must be let go. Thus saving a great cost to the company by reducing the amount of resources spent in replacing unqualified applicants and allowing the store managers to redirect that time used to other significant areas of the business.

The A.I.M system will provide prospecting applicants the ability to review open positions and their locations within that current metro area market. Through the A.I.M. system the applicant has the ability to apply for a position. When they choose to apply, the A.I.M. system will present the applicant with a qualifying questionnaire for that specific position. The A.I.M. system will automatically screen the applicant by determining if they have pass the qualifying questionnaire. The A.I.M. system will handle the applicant accordingly by providing them with a decline message if they are determined unqualified or having them create/login to an application system to complete and submit. After the applicant has completed the application with no errors, the A.I.M. system will direct them to use the phone (that AES will provide) next to the kiosk and contact a Hiring Specialist at AES's centralized hiring department.

When the Hiring Specialist has connected with the applicant they will be able to login to the A.I.M. Administrative system to find the applicant's application within a list of active applications that have passed the qualifying questionnaire and submitted a completed application. In the A.I.M. Administrative system the Hiring Specialist will be able to view applicant's digital application, be presented with the questions to conduct a phone interview, add notes from each question asked to the applicant's digital application, and then determine if they should be passed on to have an in person interview with the Hiring Manager or be decline for the position after the in person interview.

The Hiring Manager will log into the A.I.M. Administrative system to see a current list of applicants that have passed the qualify questionnaire and have been recommended by the Hiring Specialist for and in person interview for the open positions within that manager's store. They also will be able to review, add additional notes, and determine if the applicant should be submitted for a reference and background check, keep them in the hiring pool, or be declined for the position.

The A.I.M. system will have strict confidentiality of applicant's personal information and all data collected will be centralized at the company's data center's database created by 5 P.O.T.

- On 1/7/2014 and develop this pre-project notebook delivered on 2/11/2014.
- On 3/18/2014 an Iteration 1 will be presented with the following functionality:
  - Administrative interface and functions complete
  - Staffing Expert feature to create/edit/delete a job list and attached questionnaires.
- On 5/5/2014 an Iteration 2 will be presented with the following functionality added:
  - To request and/or approve open jobs.

  - The Applicant able to view jobs, apply, complete qualifying questionnaire, and submit application.
- On 6/9/2014 an Iteration 3 will be presented with the following functionality added:
  - Hiring Specialist phone interview feature and functionality.
  - Hiring Manager in person feature and functionality is complete
  - Hiring Specialist reference and background feature and functionality is complete
  - Store Manager functionality to hire Managers
  - The basic functionality for an applicant to complete application, have interviews, and finally become hired will be complete

Project Plan

A.I.M. Overview

Automated Interview Manager (A.I.M.) project is an automated web based kiosk system that replace AES's current manual job applicant screening process and incorporates a centralized hiring department within AES. The A.I.M system will remove the significant amount of time AES retail managers use to screen unqualified applicants and reduce the frequency that unqualified applicants are hired and then must be let go. Thus saving a great cost to the company in reducing the amount of resources spent in replacing unqualified applicants and allowing the store managers to redirect that time used to other significant areas of the business.

The A.I.M system provides prospecting applicants the ability to review open positions and their locations within that current metro area market. Through the A.I.M. system the applicant has the ability to apply for a position. When they apply, the A.I.M. system present the applicant with a qualifying questionnaire for that specific position. The A.I.M. system automatically screens the applicant and determines if they pass the qualifying questionnaire. The A.I.M. system handles the applicant accordingly by providing them with a decline message if they are determined unqualified or having them create/login to A.I.M.'s application system to complete and submit. After the applicant completes the application with no errors, the A.I.M. system then directs them to use the phone (that AES provides) next to the kiosk and contact a Hiring Specialist at AES's centralized hiring department.

When the Hiring Specialist has connected with the applicant they are able to login to the A.I.M. Administrative system to find the applicant's application within a list of active applications that have passed the qualifying questionnaire and submitted a completed application. The Hiring Specialist is able to view applicant's digital application, be presented with the questions to conduct the phone interview, add notes from each question asked to the applicant's digital application, and then determine if they will be passed on to have an in person interview with the Hiring Manager or be decline for the position.

The Hiring Manager logs into the A.I.M. Administrative system to see a current list of applicants that have passed the qualify questionnaire, submitted a completed application, and have been recommended by the Hiring Specialist for and in person interview for the open positions within that manager's store. They also will be able to review, add additional notes, and determine if the applicant should be submitted for a reference and background check, keep them in the hiring pool, or be declined for the position after the in person interview.

The A.I.M. system has an interface for the Staffing Expert allowing them the ability to create/edit/delete jobs, descriptions for those jobs, the qualifying questionnaire questions for those jobs, and the phone interview questions for those jobs.

The A.I.M. system has addition feature within the Hiring Manager's interface to request approval for job positions to be posted as open to their Store Manager.

The A.I.M. system has an interface for the Store Manager that provides them with the following features; to approving/declining job positions posting request from Hiring Manager, post

management position openings, view a list of applicants that have applied for management that are in the recommend for in person interview of the screening process, add additional notes to the digital application, and determine if the applicant should be submitted for a reference and background check, keep them in the hiring pool, or be declined for the position after the in person interview.

The A.I.M. system has strict confidentiality of applicant's personal information and all data collected will be centralized at the company's data center's database created by 5 P.O.T.

The A.I.M. system has a System Administrator interface that has the ability to edit login profiles and permissions. System Administrator has all the functionalities of all roles within the A.I.M. system.

A.I.M. Goals

Name: "5 P.O.T. Goals for A.I.M. Project"
Purpose: Communicate team's goals for A.I.M. Project
Team: Chase Marcum, John Price, Jordan Ringo, Josh Hartwell, Patrick Jarofski
Goals:

- On 1/7/2014 and develop this pre-project notebook delivered on 2/11/2014 with the following sections complete:
  - Title Page, Contents, Executive Summary, Project Plan, Status Reports, Requirements, Domain Model, Storyboards, Architecture, QA Plan
- On 3/18/2014 an Iteration 1 will be presented with the following requirements features complete:
  - Basic Architecture framework complete.
  - Administrative UI and functionality complete.
  - User login system complete
  - Staffing Expert functionality to create/edit/delete job list, create a qualifying questionnaire, and create a phone interview questionnaire system complete
  - Following Use cases complete - Architecture 1st Iteration, Architecture/Admin UI, SA1, SA1 Alt1, SA2, SA3, SA4, SE1, SE2, SE3, SE4, SE5, NF1, NF5, NF4
- On 5/5/2014 an Iteration 2 will be presented with the following requirements features added:
  - Reevaluating and adjusting plan according to accomplishments of iteration 1
  - Hiring Manager and Store Manager's functionality to request and approve open jobs to post complete.
  - Complete Applicant's functionality to view jobs, apply, complete qualifying questionnaire, and submit application complete.
  - Initial UI for hiring Specialist is complete.
  - Following Use cases complete – HR1 MF, SM1, A1, A2, A3, A4, A4 Alt1, A5, HS1, HS2
- On 6/9/2014 an Iteration 3 will be presented with the following requirements features added:
  - Reevaluating and adjusting plan according to accomplishments of iteration 2
  - Hiring Specialist phone interview feature and functionality complete.
  - Hiring Manager in person feature and functionality is complete.
  - Hiring Specialist reference and background feature and functionality is complete.
  - Add in Store Manager's functionality to hire Managers.
  - Complete basic functionality for an applicant process to complete application, have interviews, and finally become hired.

- Following Use cases complete – HS4, HS5, HM2, A6, HM3 MF, A7, HS3, A9, SM2, NF3, A7 Alt1, A9 Alt1, HS6, SM4

Iteration 1 Plan

Name: "A.I.M. Project Plan".
Purpose: Describe what the project is to accomplish
Living: Created during pre-project, updated each iteration during assessment phase.
Sections:

1. Overview
   Through Iteration 1 5 P.O.T plan on initializing and completing the Architecture frame for the project, develop the Administrative website's UI and functionality, implement User login system, and develop functionality to create/edit/delete users and job list with the attached questionnaire system complete.

2. goals – what the project aims to accomplish
   - Architecture frame completed.
   - Administrative UI and functionality complete.
   - User login system created.
   - Staffing Expert functionality to create/edit/delete job list and attached questionnaire system complete.

3. scope – what will be done, to what degree and what won't be done, include list of prioritized/estimated requirements
   - **√Architecture 1st Iteration - Design and implement the Database, create basic DB models, create ASP.NET MVC application, create WCF services, create interface models, and create relationships and connect all services**
   - **√Architecture / Admin UI - Designing and implementing the administrative UI that all actors will use portions of**
   - **√SA1 System Admin adds new employee to system**
   - **√SA1 Alt1 System Admin removes employee from system**
   - **√SA2 System Admin edits employee information**
   - ~~SA3 System Admin changes permission level for employee~~
   - ~~SA4 System Admin fixes typo in questionnaire~~
   - **√SE1 Staffing expert creates job title and description**
   - ~~SE2 Staffing expert creates questions to be used in questionnaires~~
   - ~~SE3 Staffing expert creates questionnaire for a job~~
   - ~~SE4 Staffing expert edits a question from pool~~
   - **√SE5 Staffing expert edits a job**
   - **√NF1 Will be used by English speaking adults**
   - **√NF5 Assumed hardware used is Computer, Keyboard, and Mouse**
   - **√NF4 Disability accommodation is currently not needed**

4. team – who is on the team, their role and time commitments

Role	Name & Contact Information	Time Commitments	Project Notebook Sections Responsible for

Team Lead Programmer	Chase Marcum chase.m.marcum@gmail.com Phone: 503-888-5604	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Overview Title Page Contents Executive Summary Management Project Plan Iteration Plan Iteration Assessment Status Report Other Supporting Documents
Requirements Lead Programmer	Patrick Jarofski pjarofski@yahoo.com Phone: 503-360-5023	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Requirements Requirements Document Domain Model Storyboards Other Supporting Documents
QA Lead Programmer	John Price john.price@oit.edu Phone: 503-896-0327	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Design QA Plan Test Cases Defect Report Other Supporting Documents
Architect Programmer	Jordan Ringo jordan.ringo@oit.edu Phone: 971-221-6447	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Design Architecture Documents
Source Control/ Documentation Specialist Programmer	Josh Hartwell josh_hartwell@msn.com Phone: 971-678-9614	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2	Deliverables Source Code

		· 7.5 days - Iteration 3	

5. time – start date, end date, other time factors (e.g. holidays)
   - Start Time - 2/11/2014 - Iteration Plan
   - 2/14/2014 - 2/17/2014 - No Jordan
   - 3/01/2014 - 3/04/2014 - No Josh
   - End Date - 3/18/2014

6. schedule – show timeline broken up into iterations, assign requirements to iterations
   - Start Time - 2/11/2014 - Iteration Plan
   - 2/15/2014 - Requirements Meeting - (No Jordan)
   - End Date - 3/18/2014

Iteration 1 Assessment

Name: "A.I.M. Iteration 1 Assessment"
Purpose: Record results of iteration
Historical: Created during assessment phase of iteration.
Sections:

1. accomplished – what was completed during the iteration
   - AIM Database
   - AIM WCF Service
   - AIM Administrative Website
     - Created user views
     - Created job views
     - Created question views
   - Create a user
   - Create a job
     a. Developed initial Administrative UI
     b. Created basic login w/o permissions
     c. Accomplished prototype for creating questions
     d. Group communication including organizing meetings in the face of obstacles
     e. Collaborate and discuss our opinions

2. left todo – what was not accomplished, or new items that need to be done
   - Create question
   i. Implement JSON logic
   - Implement permission levels
   - Fix editing bug
   - Testing

3. adjustments – changes required to project scope or schedule
   - Permissions pushed to next iteration
   - Creating questions pushed to next iteration
   - Uses cases pushed to next iteration:
     - SA3 System Admin changes permission level for employee
     - SA4 System Admin fixes typo in questionnaire
     - SE2 Staffing expert creates questions to be used in questionnaires
     - SE3 Staffing expert creates questionnaire for a job
     - SE4 Staffing expert edits a question from pool

4. improvements – changes to process and architecture identified by team
   - Now that the architecture is in place, workload can be distributed better.
   - More experience with Git source control.
   - We need to work on isolating work that can be accomplished in parallel.
   - We need to work on implementing a method for mass unit testing and integration testing.

Iteration 2 Plan

Name: "A.I.M. Project Plan".
Purpose: Describe what the project is to accomplish
Living: Created during pre-project, updated each iteration during assessment phase.
Sections:

1. Overview

   Through Iteration 2 5 P.O.T plans on finishing the Admin Web features and User Interface, design and develop the Applicant User Interface, and create the ability for the applicant to apply for a job.

2. goals – what the project aims to accomplish
   - Finish Admin features and UI
   - Create Applicant UI
   - Create the ability to apply for a job

3. scope – what will be done, to what degree and what won't be done, include list of prioritized/estimated requirements
   - P1 Implement Unit testing throughout iteration
   - P2 Finish implementing user log in
   - P3 Database Validation of information
   - P4 Implement JSon Logic for questions
   - SA2 System Admin edits employee information
   - SE3 Staffing expert creates questionnaire for a job
   - A1 Applicant chooses a job to apply for
   - A2 Applicant creates a user account with AES
   - A3 Applicant completes questionnaire
   - A4 Applicant fills out and submits applications after passing questionnaire
   - A4 Alt1 Applicant does not pass questionnaire and applies for a different job
   - A5 Applicant calls HS for a phone interview (See HS4)

4. team – who is on the team, their role and time commitments

Role	Name & Contact Information	Time Commitments	Project Notebook Sections Responsible for
Team Lead Programmer	Chase Marcum chase.m.marcum@gmail.com Phone: 503-888-5604	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2	Overview Title Page Contents Executive Summary Management Project Plan

		· 7.5 days - Iteration 3	Iteration Plan Iteration Assessment Status Report Other Supporting Documents
Requirements Lead Programmer	Patrick Jarofski pjarofski@yahoo.com Phone: 503-360-5023	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Requirements Requirements Document Domain Model Storyboards Other Supporting Documents
QA Lead Programmer	John Price john.price@oit.edu Phone: 503-896-0327	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Design QA Plan Test Cases Defect Report Other Supporting Documents
Architect Programmer	Jordan Ringo jordan.ringo@oit.edu Phone: 971-221-6447	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Design Architecture Documents
Source Control/ Documentation Specialist Programmer	Josh Hartwell josh_hartwell@msn.com Phone: 971-678-9614	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Deliverables Source Code

5. time – start date, end date, other time factors (e.g. holidays)

- Start Time - 4/1/2014 - Iteration 2 Plan

- 4/4/2014 - 4/6/2014 - No Chase
- 5/10/2014 - 5/17/2014 - No Jordan (Hawaii!!!!!!!!!!!!)
- End Date -5/6/2014

6. schedule – show timeline broken up into iterations, assign requirements to iterations
   - Week 1:
     - Planning Requirements
   - Week 2:
     - Design
   - Week 3:
     - Code
   - Week 4:
     - Code
   - Week 5:
     - Integration/System Testing

Iteration 2 Assessment

Name: "A.I.M. Iteration 1 Assessment"
Purpose: Record results of iteration
Historical: Created during assessment phase of iteration.
Sections:

1. accomplished – what was completed during the iteration
   - Create a client-side application
     - Start page view to select Region
     - Open Job list by region view
     - Open Job details view
     - Created Open Job Controller
     - Create Application View
   - Application Services created
     - ApplicationService Created
     - JobService Created
     - QuestionService Created
   - Unit tests for data created
   - Unit tests for services and controllers partially implemented
   - Planned scope for Iteration 2
     - P1 Implement Unit testing throughout iteration
     - ~~P2 Finish implementing user log in~~
     - ~~P3 Database Validation of information~~
     - P4 Implement JSon Logic for questions
     - ~~SA2 System Admin edits employee information~~
     - ~~SE3 Staffing expert creates questionnaire for a job~~
     - A1 Applicant chooses a job to apply for
     - ~~A2 Applicant creates a user account with AES~~
     - ~~A3 Applicant completes questionnaire~~
     - A4 Applicant fills out ~~and submits~~ applications after passing questionnaire
     - ~~A4 Alt1 Applicant does not pass questionnaire and applies for a different job~~
     - ~~A5 Applicant calls HS for a phone interview (See HS4)~~
2. left todo – what was not accomplished, or new items that need to be done
   - Create view for questionnaires, applications, and creating an applicant account.
   - Finish adding a Login feature that uses roles (permission) and is tied directly to the database.
   - Additional Testing.
   - The team work to support the entire team when multiple personal emergencies occurred during this iteration.
   - Finish partially done use cases
     - P2 Finish implementing user log in
     - P3 Database Validation of information
     - A2 Applicant creates a user account with AES
     - A3 Applicant completes questionnaire
     - A5 Applicant submits application
     - Not implemented use cases
     - SA2 System Admin edits employee information

- SE3 Staffing expert creates questionnaire for a job
- A4 Alt1 Applicant does not pass questionnaire and applies for a different job
- A5 Applicant calls HS for a phone interview (See HS4)

3. adjustments – changes required to project scope or schedule
   - Permissions pushed to next iteration
   - Creating questionnaires pushed to next iteration
   - Uses cases pushed to next iteration:
     - Finish partially done use cases
       - P2 Finish implementing user log in
       - P3 Database Validation of information
       - A2 Applicant creates a user account with AES
       - A3 Applicant completes questionnaire
       - A5 Applicant submits application
     - Not implemented use cases
       - SA2 System Admin edits employee information
       - SE3 Staffing expert creates questionnaire for a job
       - A4 Alt1 Applicant does not pass questionnaire and applies for a different job
       - A5 Applicant calls HS for a phone interview (See HS4)
4. improvements – changes to process and architecture identified by team
   - More experience with Git source control.
   - We need to work on isolating work that can be accomplished in parallel.
   - We need to work on implementing a method for integration testing.
   - Pair programming to implement testing.
   - More frequent communication of status and/or issues that become apparent.
   - Working together to help overcome roadblocks that arise.

Iteration 3 Plan

Name: "A.I.M. Project Plan".
Purpose: Describe what the project is to accomplish
Living: Created during pre-project, updated each iteration during assessment phase.
Sections:

1. Overview
   Through Iteration 1 5 P.O.T plan on initializing and completing the Architecture frame for the project, develop the Administrative website's UI and functionality, implement User login system, and develop functionality to create/edit/delete users and job list with the attached questionnaire system complete.

2. goals – what the project aims to accomplish
   - Having a functional Application Website complete.
   - Have a well-planned presentation.

3. scope – what will be done, to what degree and what won't be done, include list of prioritized/estimated requirements
   - P2 Finish implementing user log in
   - P3 Database Validation of information
   - SA2 System Admin edits employee information
   - SE3 Staffing expert creates questionnaire for a job
   - A2 Applicant creates a user account with AES
   - A3 Applicant completes questionnaire
   - A4 Alt1 Applicant does not pass questionnaire and applies for a different job
   - A5 Applicant calls HS for a phone interview (See HS4)

4. team – who is on the team, their role and time commitments

Role	Na & Contact Information	Time Commitments	Project Notebook Sections Responsible for
Team Lead Programmer	Chase Marcum chase.m.marcum@gmail.com Phone: 503-888-5604	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Overview Title Page Contents Executive Summary Management Project Plan Iteration Plan Iteration Assessment Status Report Other Supporting Documents

Requirements Lead Programmer	Patrick Jarofski pjarofski@yahoo.com Phone: 503-360-5023	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Requirements Requirements Document Domain Model Storyboards Other Supporting Documents
QA Lead Programmer	John Price john.price@oit.edu Phone: 503-896-0327	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Design QA Plan Test Cases Defect Report Other Supporting Documents
Architect Programmer	Jordan Ringo jordan.ringo@oit.edu Phone: 971-221-6447	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Design Architecture Documents
Source Control/ Documentation Specialist Programmer	Josh Hartwell josh_hartwell@msn.com Phone: 971-678-9614	· 1.5 days of work per week · 7.5 days - Iteration 1 · 7.5 days - Iteration 2 · 7.5 days - Iteration 3	Deliverables Source Code

5. time – start date, end date, other time factors (e.g. holidays)
   - Start Time - 4/1/2014 - Iteration 2 Plan
   - 5/10/2014 - 5/17/2014 - No Jordan (Hawaii!!!!!!!!!!!)!
   - End Date -6/6/2014
6. schedule – show timeline broken up into iterations, assign requirements to iterations

- Week 1:
  - Planning Requirements, Design & Code
- Week 2:
  - Code & Unit Testing
- Week 3:
  - Code & Unit Testing
- Week 4:
  - Code & Unit Testing
- Week 5:
  - Create and Prepare for Presentation & Integration/System Testing

Iteration 3 Assessment

Name: "A.I.M. Iteration 1 Assessment"
Purpose: Record results of iteration
Historical: Created during assessment phase of iteration.
Sections:

1. Accomplished – what was completed during the iteration
   - Created a client-side application
     - Integrated a GUI (css and js) for stepping through data on a single page
     - Created Questionnaire View
     - Created Application View
   - Created an admin-side application
     - Browse job openings
       - Create new openings
       - Delete existing ones
       - Approve openings that are pending approval
     - View, Delete, and Edit Questions.
     - Alert messages display for create, edit, delete
     - Implemented permissions for all actors
   - Implemented working services
     - Administrative service
     - Application service
     - Deployed both
     - Testing tool for service used
   - Unit tests for data created
   - Unit tests for services and controllers partially implemented
   - Planned scope for Iteration 3
     - P1 Implement Unit testing throughout iteration
     - P2 Finish implementing user log in with roles / permissions
     - ~~P3 HTML Validation of information~~
     - SA2 System Admin edits employee information

- SE4 Staffing expert edits a question from pool
- AP3 Applicant completes questionnaire
- AP4 Applicant fills out and submits applications after passing questionnaire
- AP4 Alt1 - Applicant does not pass questionnaire and applies for a different job
- AP5 Applicant calls HS for a phone interview (See HS4)
- HM1 Hiring Manager requests a job opening be approved
- SM1 Store Manager approves an unapproved job opening

2. left todo – what was not accomplished, or new items that need to be done
   - Company lost funding and project is canceled. (End of Junior Project)

3. adjustments – changes required to project scope or schedule
   - Not implemented use cases
     - SA2 System Admin edits employee information
     - SE3 Staffing expert creates questionnaire for a job
     - P3 HTML validation of input.
     - AP2 Applicant creates a user account.
     - AP5 Applicant Calls HS for phone interview.
   - Planned Out of Scope for project.
     - SA3 System Admin Changes permission level for an employee.
     - HS1 Hiring Specialist views applicants for a specific job opening.
     - HS2 Hiring Specialist views an applicant's information

4. improvements – changes to process and architecture identified by team
   - More experience with Git source control.
   - We need to work on isolating work that can be accomplished in parallel.
   - We need to work on implementing a method for integration testing.
   - Pair programming to implement testing.
   - More frequent communication of status and/or issues that become apparent.
   - Working together to help overcome roadblocks that arise.

A.I.M. Scope

Requirements Allocation to Iterations

ID#	Use Case #	Description	Need	Risk	Time Est. (In Days)	Week planned
50	Architecture 1st Iteration	Design and implement the Database, create basic DB models, create ASP.NET MVC application, create WCF services, create interface models, and create relationships and connect all services	High	High	~~15~~ 25	Week 7 & 8
51	Architecture / Admin UI	Designing and implementing the administrative UI that all actors will use portions of	High	High	2.5	
16	SA1	System Admin adds new employee to system	High	Low	1.5	
17	SA1 Alt1	System Admin removes employee from system	Low	Low	1.5	
~~18~~	~~SA2~~	~~System Admin edits employee information~~	~~High~~	~~Low~~	~~1.5~~	-
~~19~~	~~SA3~~	~~System Admin changes permission level for employee~~	~~Med~~	~~Low~~	~~1.5~~	~~Week 9~~
~~20~~	~~SA4~~	~~System Admin fixes typo in questionnaire~~	~~Low~~	~~Low~~	~~2~~	-
35	SE1	Staffing expert creates job title and description	High	Low	1.5	
36	SE2	Staffing expert creates questions to be used in questionnaires	High	Low	1.5	
~~37~~	~~SE3~~	~~Staffing expert creates questionnaire for a job~~	~~High~~	~~Med~~	~~3~~	~~Week 10~~
~~38~~	~~SE4~~	~~Staffing expert edits a question from pool~~	~~Low~~	~~Low~~	~~2~~	-
39	SE5	Staffing expert edits a questionnaire for a job	Low	Med	4	
45	NF1	Will be used by English speaking adults	High	Low	0	
49	NF5	Assumed hardware used is Computer, Keyboard, and Mouse	High	Low	0	
48	NF4	Disability accommodation is currently not needed	Low	Low	0	Week 11
		End of Iteration 1			**37.5**	
	P1	Implement Unit testing throughout iteration	High	Med	7.5	
	~~P2~~	~~Finish implementing user log in~~	~~High~~	~~Low~~	~~2~~	
	~~P3~~	~~Database/Html validation of information~~	~~High~~	~~Low~~	~~2~~	
	P4	Implement Json Logic for questions	High	Med	3	
~~18~~	~~SA2~~	~~System Admin edits employee information~~	~~High~~	~~Low~~	~~1.5~~	-
~~37~~	~~SE3~~	~~Staffing expert creates questionnaire for a job~~	~~High~~	~~Med~~	~~3~~	
1	A1	Applicant chooses a job to apply for	High	High	15	
~~2~~	~~A2~~	~~Applicant creates an user account with AES~~	~~High~~	~~Low~~	~~1.5~~	-
~~3~~	~~A3~~	~~Applicant completes questionnaire~~	~~High~~	~~Med~~	~~4~~	-
4	A4	Applicant fills out and submits applications after passing questionnaire	High	Med	4	
~~5~~	~~A4 Alt1~~	~~Applicant does not pass questionnaire and applies for a different job~~	~~High~~	~~Low~~	~~1.5~~	-

~~7~~	~~A5~~	~~Applicant calls HS for a phone interview (See HS4)~~	~~High~~	~~Low~~	~~0.5~~	-
		End of Iteration 2			**37.5**	
	P1	Implement Unit testing throughout iteration	High	Med	9	
	P2	Finish implementing user log in	High	Low	3	
	P3	Database/Html validation of information	High	Low	3	
18	SA2	System Admin edits employee information	High	Low	1.5	
37	SE3	Staffing expert creates questionnaire for a job	High	Med	3	
2	A2	Applicant creates an user account with AES	High	Low	1.5	
3	A3	Applicant completes questionnaire	High	Med	4	
5	A4 Alt1	Applicant does not pass questionnaire and applies for a different job	High	Low	1.5	
7	A5	Applicant calls HS for a phone interview (See HS4)	High	Low	0.5	
20	SA4	System Admin fixes typo in questionnaire	Low	Low	0.5	
38	SE4	Staffing expert edits a question from pool	Low	Low	3	
40	HS1	Hiring Specialist views list of applicants for a specific job listing	High	Med	3	
41	HS2	Hiring Specialist views an applicant's information	High	Med	3	
		End of Iteration 3			**36**	

----	----------	---------- Cut Line (Out of Scope) ----------	-------	-------	--------	----------
34	HM2	Hiring Manager reviews applications	High	Med	7	
9	A6	Applicant has in-person interview after passing the phone interview	High	Low	0.5	
29	HM3 MF	Hiring Manager conducts in person interview and recommends to hire	High	Med	4	
11	A7	Applicant gets recommended for Hire after in-person interview	High	Low	0.5	
42	HS3	Hiring Specialist checks applicant references and does background check	High	Med	4	
14	A9	Applicant is hired after background and reference checks are passed	High	Low	0.5	
23	SM2	Store Manager creates openings for a management position.	High	Med	2	
47	NF3	Allowed downtime is when no stores are open	High	Med	5	
12	A7 Alt1	Applicant does not pass in-person interview and is removed from hiring pool	High	Low	0	

15	A9 Alt1	Applicant does not pass background and reference checks, and is removed from hiring pool (HS)	High	Low	0	
~~52~~	~~HS6~~	~~Hiring Specialist removes applicant after failing background/reference checks~~	~~High~~	~~Low~~	~~0.5~~	
~~26~~	~~SM4~~	~~Store Manager acts as Hiring Manager, with Hiring Manager permissions, for interviews for management positions.~~	~~Med~~	~~Med~~	4	
52	HS6	Hiring Specialist removes applicant after failing background/reference checks	High	Low	0.5	
26	SM4	Store Manager acts as Hiring Manager, with Hiring Manager permissions, for interviews for management positions.	Med	Med	4	
32	HR2 Alt 3	Hiring Manager calls to schedule and is unable to contact applicant	Low	Low	2	
33	HR2 Alt4	Applicant declines the position when called	Low	Low	2	
25	SM3	Store Manager accesses position list and reviews position details.	Low	Low	1.5	
6	A4 Alt2	Applicant does not pass questionnaire and doesn't apply for other jobs	Low	Low	1	
10	A6 Alt1	Applicant does not pass phone interview and is removed from hiring pool	Low	Low	0.5	
28	HR1 Alt1	Hiring Manager requests positions to close	Low	Low	2	
13	A8	Applicant is subject to background and reference checks	Low	Low	0	
30	HR2 Alt1	Hiring Manager conducts in person interview and declines	Low	Med	4	
31	HR2 Alt2	Hiring Manager conducts in person interview and places in pool	Low	Med	4	
22	SM1 Alt1	Store Manager accesses position list and declines a position.	Low	Med	3	
24	SM2 Alt1	Store Manager removes openings for a management position.	Low	Med	3	

TABLE 1 – A.I.M. REQUIREMENTS ALLOCATION TO ITERATIONS

5 P.O.T. Team

5 P.O.T. ROLES & RESPONSIBILITIES

Project A.I.M. - Automated Interview Manager

Role	Name & Contact Information	Time Commitments	Project Notebook Sections Responsible for
Team Lead Programmer	**Chase Marcum** chase.m.marcum@gmail.com Phone: 503-888-5604	• 1.5 days of work per week • 7.5 days - Iteration 1 • 7.5 days - Iteration 2 • 7.5 days - Iteration 3	**Overview** Title Page Contents Executive Summary **Management** Project Plan Iteration Plan Iteration Assessment Status Report Other Supporting Documents
Requirements Lead Programmer	**Patrick Jarofski** pjarofski@yahoo.com Phone: 503-360-5023	• 1.5 days of work per week • 7.5 days - Iteration 1 • 7.5 days - Iteration 2 • 7.5 days - Iteration 3	**Requirements** Requirements Document Domain Model Storyboards Other Supporting Documents
QA Lead Programmer	**John Price** john.price@oit.edu Phone: 503-896-0327	• 1.5 days of work per week • 7.5 days - Iteration 1 • 7.5 days - Iteration 2 • 7.5 days - Iteration 3	**Design** Architecture Documents
Architect Programmer	**Jordan Ringo** jordan.ringo@oit.edu Phone: 971-221-6447	• 1.5 days of work per week • 7.5 days - Iteration 1 • 7.5 days - Iteration 2 • 7.5 days - Iteration 3	**QA** QA Plan Test Cases Defect Report Other Supporting Documents
Source Control/ Documentation Specialist Programmer	**Josh Hartwell** josh_hartwell@msn.com Phone: 971-678-9614	• 1.5 days of work per week • 7.5 days - Iteration 1 • 7.5 days - Iteration 2 • 7.5 days - Iteration 3	**Deliverables** Source Code

TABLE 2 - 5 P.O.T ROLES & RESPONSIBILITIES

A.I.M. Project Timeline

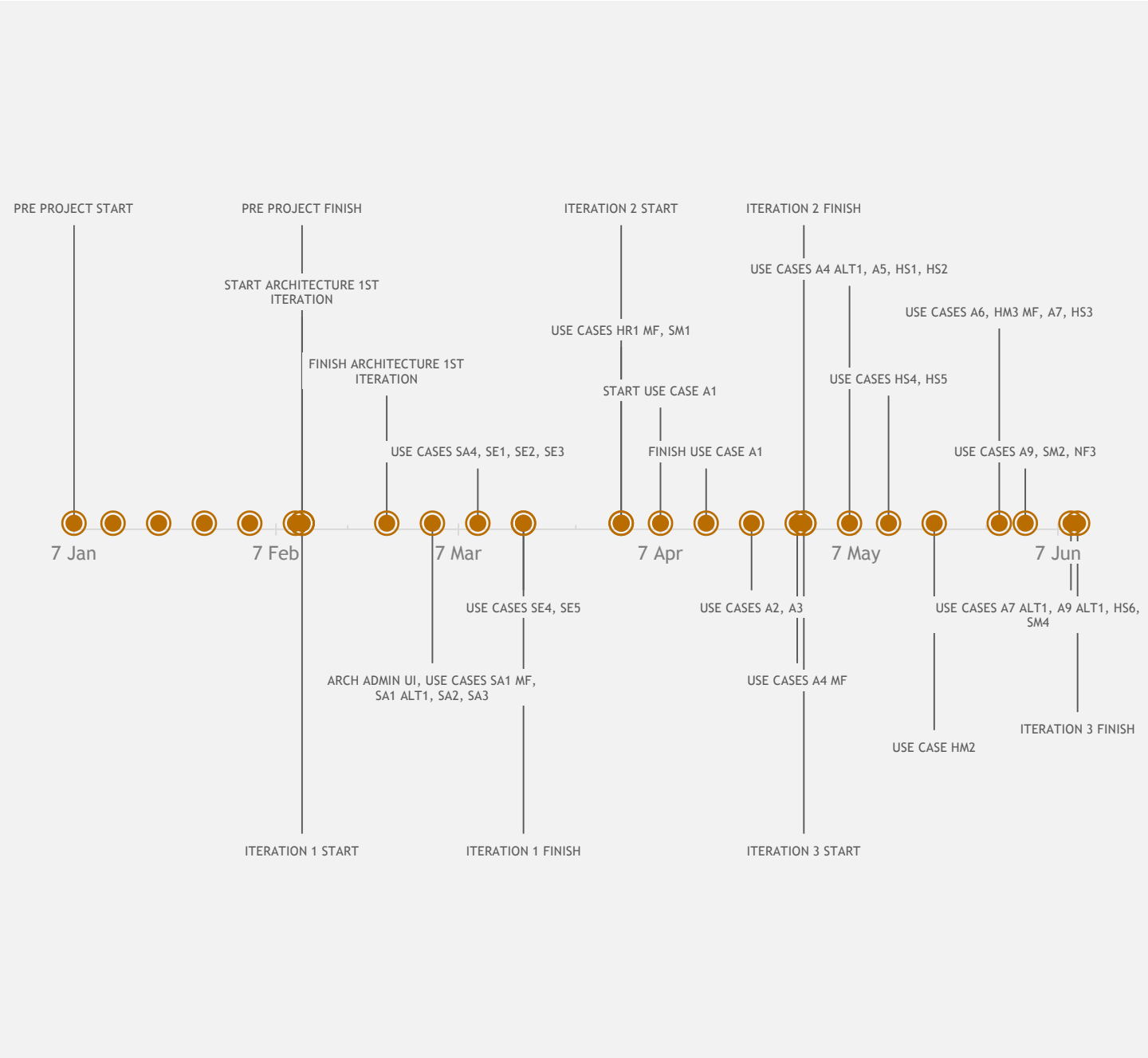


TABLE 3 - A.I.M. PROJECT TIMELINE

A.I.M. Project Schedule

A.I.M. PROJECT PLAN SCHEDULE

PROJECT NAME	**A.I.M.** AUTOMATED INTERVIEW MANAGER	**Team Lead** - Chase Marcum **Requirements Lead** - Patrick Jarofski **QA Lead** - John Price **Architect** - Jordan Ringo **Source Control/Documentation Specialist** - Josh Hartwell
PROJECT TEAM	**5 P.O.T.** 5 PROGRAMMERS OF TOMORROW	

WEEK	STARTING	ENDING	WEEK GOALS
Pre-Project Period			
WEEK 1	1.7.2014	1.13.2014	• Prepare Questions for Requirements Interview • Define Roles & Responsibilities • Set up Document & Program Repository • Create Preliminary Tentative TODO Schedule • Create Agenda for Week 2 meeting • Status 1 - Chase

WEEK 2	1.14.2014	1.20.2014	• Requirements Team Meeting
			  ○ Understand current Business Process
			  ○ Understand expected new Business Process
			  ○ Identify Actors
			  ○ Identify Use Cases w/ brief descriptions
			  ○ Identify Non-functional Requirements w/ brief descriptions
			  ○ Create Storyboards for Complex Use Cases
			• Complete Initial Requirements Document
			• Complete Initial Domain Model
			• Complete Initial Storyboards
			• Create Agenda for Week 3 meeting
			• Status 2 – Patrick

WEEK 3	1.21.2014	1.27.2014	• Architecture Team Meeting
			  ○ Propose a set of components
			  ○ Identify architecturally significant use cases
			  ○ Analyze each identified scenario
			  ○ Unify the components & interactions
			  ○ Choose technology for each component
			• Complete Initial Architecture Document

			• Complete Initial Complete Design Document • Create Agenda for Week 4 meeting • Status 3 – John

WEEK 4	1.28.2014	2.3.2014	• QA Team Meeting
			  ○ Create QA high level description overview
			  ○ Create QA high level description of testing priorities
			  ○ Develop QA scope
			  ○ Identify QA testing process
			• Complete Initial QA Plan
			• Complete Initial Test Cases
			• Complete Initial Defect Report
			• Create Agenda for Week 5 meeting
			• Status 4 – Jordan

WEEK 5	2.4.2014	2.10.2014	• Planning Team Meeting
			  ○ Create Requirements List
			  ○ Identify Needs
			  ○ Identify Risks
			  ○ Identify Prioritizes
			  ○ Create Estimates
			  ○ Amend Scheduled Details
			• Complete Initial Project Plan Document
			• Complete Initial Executive Summary
			• Create Agenda for Week 6 meeting
			• Status 5 – Josh

Iteration 1

WEEK 6	2.11.2014	2.17.2014	• Iteration 1 Team Meeting
			  ○ Develop a iteration 1 plan
			• Complete Iteration 1 Plan Document
			• Requirements
			  ○ Begin Architecture 1st Iteration
			• Create Agenda for Week 7 meeting
			• Status 6 - Chase
			• Pre-Project Notebook completed

WEEK 7	2.18.2014	2.24.2014	• Create Agenda for Week 8 meeting • Requirements

			○ Finish Architecture 1st Iteration • Status 7 – Patrick
WEEK 8	2.25.2014	3.3.2014	• Create Agenda for Week 9 meeting • Requirements ○ Architecture Admin UI ○ Use Case SA1 MF ○ Use Case SA1 Alt1 ○ Use Case SA2 ○ Use Case SA3 • Status 8 – John
WEEK 9	3.4.2014	3.10.2014	• Create Agenda for Week 10 meeting • Requirements ○ Use Case SA4 ○ Use Case SE1 ○ Use Case SE2 ○ Use Case SE3 • Status 9 – Jordan
WEEK 10	3.11.2014	3.17.2014	• Create Agenda for 30-45 minute Iteration 1 Presentation • Requirements ○ Use Case SE4 ○ Use Case SE5 • Prepare Iteration 1 Presentation • Status 10 - Josh
WEEK 11	3.18.2014	3.18.2014	• Present Iteration 1 Presentation • Status 11 – Chase
WEEK 12	3.18.2014	3.31.2014	**Spring Break – Office Closed**
Iteration 2			
WEEK 13	4.1.2014	4.7.2014	• Create Agenda for Week 14 meeting • Iteration 2 Phase – Planning ○ Develop Iteration 2 Plan ○ Update Schedule • Iteration 2 Phase - Requirement Planning ○ P1 Implement Unit testing throughout iteration ○ P2 Finish implementing user log in ○ P3 Database Validation of information ○ P4 Implement JSon Logic for questions

			○ SA2 System Admin edits employee information ○ SE3 Staffing expert creates questionnaire for a job ○ A1 Applicant chooses a job to apply for ○ A2 Applicant creates an user account with AES ○ A3 Applicant completes questionnaire ○ A4 Applicant fills out and submits applications after passing questionnaire ○ A4 Alt1 Applicant does not pass questionnaire and applies for a different job ○ A5 Applicant calls HS for a phone interview (See HS4) • Status 12 – Patrick
WEEK 14	4.8.2014	4.14.2014	• Create Agenda for Week 15 meeting • Iteration 2 Phase – Design ○ P1 Implement Unit testing throughout iteration ○ P2 Finish implementing user log in ○ P3 Database Validation of information ○ P4 Implement JSon Logic for questions ○ SA2 System Admin edits employee information ○ SE3 Staffing expert creates questionnaire for a job ○ A1 Applicant chooses a job to apply for ○ A2 Applicant creates an user account with AES ○ A3 Applicant completes questionnaire ○ A4 Applicant fills out and submits applications after passing questionnaire ○ A4 Alt1 Applicant does not pass questionnaire and applies for a different job ○ A5 Applicant calls HS for a phone interview (See HS4) • Status 13 – Chase
WEEK 15	4.15.2014	4.21.2014	• Create Agenda for Week 16 meeting • Iteration 2 Phase – Code week 1 of 2 ○ P1 Implement Unit testing throughout iteration ○ P2 Finish implementing user log in ○ P3 Database Validation of information ○ P4 Implement JSon Logic for questions ○ SA2 System Admin edits employee information ○ SE3 Staffing expert creates questionnaire for a job ○ A1 Applicant chooses a job to apply for ○ A2 Applicant creates an user account with AES ○ A3 Applicant completes questionnaire ○ A4 Applicant fills out and submits applications after passing questionnaire

			○ A4 Alt1 Applicant does not pass questionnaire and applies for a different job ○ A5 Applicant calls HS for a phone interview (See HS4) • Status 14 – John
WEEK 16	4.22.2014	4.28.2014	• Create Agenda for Week 17 meeting • Iteration 2 Phase – Code week 2 of 2 ○ P1 Implement Unit testing throughout iteration ○ P2 Finish implementing user log in ○ P3 Database Validation of information ○ P4 Implement JSon Logic for questions ○ SA2 System Admin edits employee information ○ SE3 Staffing expert creates questionnaire for a job ○ A1 Applicant chooses a job to apply for ○ A2 Applicant creates an user account with AES ○ A3 Applicant completes questionnaire ○ A4 Applicant fills out and submits applications after passing questionnaire ○ A4 Alt1 Applicant does not pass questionnaire and applies for a different job ○ A5 Applicant calls HS for a phone interview (See HS4) • Status 15 – Jordan
WEEK 17	4.29.2014	5.5.2014	• Create Agenda for Week 18 meeting • Iteration 2 Phase – Test • Iteration 2 Phase – Assess • Status 16 – Josh

Iteration 3

WEEK 18	5.6.2014	5.12.2014	• Create Agenda for Week 19 meeting • Iteration 3 Phase – Planning ○ Develop Iteration 2 Plan ○ Update Schedule • Iteration 3 Phase - Requirement Planning • Iteration 3 Phase – Design ○ P2 Finish implementing user log in ○ P3 Database Validation of information ○ SA2 System Admin edits employee information

			○ SE3 Staffing expert creates questionnaire for a job ○ A2 Applicant creates an user account with AES ○ A3 Applicant completes questionnaire ○ A4 Alt1 Applicant does not pass questionnaire and applies for a different job ○ A5 Applicant calls HS for a phone interview (See HS4) • Status 17 – Chase
WEEK 19	5.13.2014	5.19.2014	• Create Agenda for Week 20 meeting • Iteration 3 Phase – Code ○ P2 Finish implementing user log in ○ P3 Database Validation of information ○ SA2 System Admin edits employee information ○ SE3 Staffing expert creates questionnaire for a job ○ A2 Applicant creates an user account with AES ○ A3 Applicant completes questionnaire ○ A4 Alt1 Applicant does not pass questionnaire and applies for a different job ○ A5 Applicant calls HS for a phone interview (See HS4) • Status 17 – John
WEEK 20	5.20.2014	5.26.2014	• Create Agenda for Week 21 meeting • Iteration 3 Phase – Code ○ P2 Finish implementing user log in ○ P3 Database Validation of information ○ SA2 System Admin edits employee information ○ SE3 Staffing expert creates questionnaire for a job ○ A2 Applicant creates an user account with AES ○ A3 Applicant completes questionnaire ○ A4 Alt1 Applicant does not pass questionnaire and applies for a different job ○ A5 Applicant calls HS for a phone interview (See HS4) • Status 18 – Jordan
WEEK 21	5.27.2014	6.2.2014	• Create Agenda for Week 22 meeting • Iteration 3 Phase – Code ○ P2 Finish implementing user log in ○ P3 Database Validation of information

			○ SA2 System Admin edits employee information ○ SE3 Staffing expert creates questionnaire for a job ○ A2 Applicant creates an user account with AES ○ A3 Applicant completes questionnaire ○ A4 Alt1 Applicant does not pass questionnaire and applies for a different job ○ A5 Applicant calls HS for a phone interview (See HS4) • Status 19 – Josh
WEEK 22	6.3.2014	6.9.2014	• Create Agenda for Week 23 meeting • Iteration 3 Phase – Test • Iteration 3 Phase – Assess • Prepare Presentation • Status 20 – Chase
	6.10.2014	6.10.2014	• Present Iteration 3 Presentation • Status 21 – Patrick

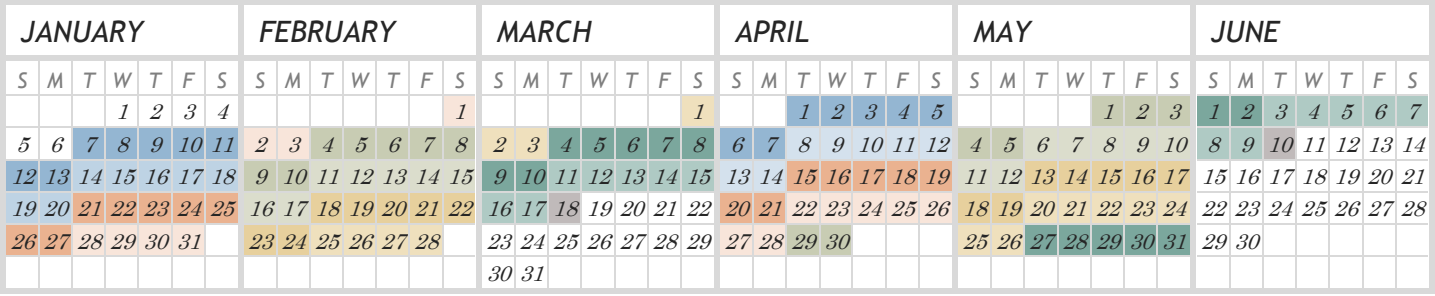


TABLE 4 - A.I.M. PROJECT PLAN SCHEDULE

Status Reports

5 P.O.T. Status Report 14 January 2014

Name: "5 P.O.T. Status Report 14 January 2014"
Purpose: Communicate team's status to project manager
Historical. Created each week.
Sections:

1. Accomplished –
   - Prepared questions for requirements interview.
   - Defined roles and responsibilities.
   - Established document and program repositories and ensured entire team has access.
   - Created preliminary tentative TODO schedule.
   - Prepared status report 1

2. Planned – what is planned for next week
   - Requirements Team Meeting
     - Understand current Business Process
     - Understand expected new Business Process
     - Identify Actors
     - Identify Use Cases w/ brief descriptions
     - Identify Non-functional Requirements w/ brief descriptions
     - Create Storyboards for Complex Use Cases
   - Complete Initial Requirements Document
   - Complete Initial Domain Model
   - Complete Initial Storyboards
   - Create Agenda for Week 3 meeting
   - Status 2 – Patrick

3. Issues – new issues that have arisen this week, what has been done for issues on previous status reports
   - Plan to network with other groups fell through due to the other group changing their mind that it was not a priority to complete.

4. Scope:
   N/A

5 P.O.T. Status Report 21 January 2014

Name: "5 P.O.T. Status Report 21 January 2014"
Purpose: Communicate team's status to project manager
Team: Chase Marcum, John Price, Jordan Ringo, Josh Hartwell, Patrick Jarofski
Sections:

7. Accomplished:
    - Asked the customer questions about requirements
    - Pooled together our notes about requirements
    - Identified Actors
    - Developed Domain Model
    - Identified Use Cases
    - Created 3 Detailed use cases
    - Created initial Requirements Document
    - Created a storyboard for one of the detailed use cases
    - Identified additional questions for the customer

8. Planned:
    - Architecture meeting
        - Propose a set of components
        - Identify architecturally significant use cases
        - Analyze each identified scenario
        - Unify the components and interactions
        - Choose technology for each component
    - Complete initial Architecture document
    - Complete initial Complete Design document
    - Create an agenda for week 3

9. Issues:
    - Ran out of time on Tuesday night, resolved by having an additional meeting Friday night

10. Scope:
    N/A

5 P.O.T. Status Report 28 January 2014

Name: "5 P.O.T. Status Report 28 January 2014"
Purpose: Communicate team's status to project manager
Team: Chase Marcum, John Price, Jordan Ringo, Josh Hartwell, Patrick Jarofski
Sections:

1. Accomplished:
   - Asked the customer questions about requirements
   - Identified Architecture(Azure cloud, asp.net MVC, MCF, SQL)
   - Created initial Architecture Document
   - Created interaction diagrams of the detailed use cases
   - Identified additional questions for the customer

2. Planned:
   - QA meeting
     - Create QA high level description
     - Create QA high level priorities
     - Develop QA scope
     - Identify a testing process
     - Complete initial QA plan and Test Cases
   - Complete initial QA document
   - Create an agenda for week 4

3. Issues:
   - Ran into a little confusion regarding requirements, will resolve privately with customer.

4. Scope:
   N/A

5 P.O.T. Status Report 4 February 2014

Name: "5 P.O.T. Status Report 4 February 2014"
Purpose: Communicate team's status to project manager
Team: Chase Marcum, John Price, Jordan Ringo, Josh Hartwell, Patrick Jarofski
Sections:

1. Accomplished –
   - QA Team Meeting - Everyone
     - Create QA high level description overview
     - Create QA high level description of testing priorities
     - Develop QA scope
     - Identify QA testing process
   - Initial QA Plan - John
   - Initial Test Cases - John
   - Initial Defect Report – John
   - Brainstormed Ideas for Object Classes and Relational Schema – Everyone
   - Researched ASP.NET Web Service Techniques – Chase
   - Clarified with customer on survey related questions – Everyone
   - Create Agenda for Week 5 meeting – Jordan

2. Planned – what is planned for next week
   - Planning Team Meeting - Everyone
     - Create Requirements List
     - Identify Needs
     - Identify Risks
     - Identify Prioritizes
     - Create Estimates
     - Amend Scheduled Details
   - Discuss Class and Relational Schema
   - Complete Initial Project Plan Document
   - Complete Initial Executive Summary
   - Create Agenda for Week 6 meeting
   - Create Status Report 5 – Josh

3. Issues – new issues that have arisen this week, what has been done for issues on previous status reports
   - No new issues arose/
   - All previous issues have been resolved/

4. Scope:
   N/A

5 P.O.T. Status Report 11 February 2014

Name: "5 P.O.T. Status Report 11 February 2014"
Purpose: Communicate team's status to project manager
Team: Chase Marcum, John Price, Jordan Ringo, Josh Hartwell, Patrick Jarofski
Sections:

1. Accomplished:
   - Planning Team Meeting - Everyone
     - Create Risk Analysis document
     - Create schedules for all three iterations
     - Identify project priorities
     - Create initial time estimates for each iteration
   - Initial High Level use cases for every actor - Everyone
   - Compile and Format all documents into the pre-project notebook – Chase
   - Create Status 5 Report - Josh
   - Create Agenda for Week 6 meeting – Chase

2. Planned:
   - Iteration 1 Team Meeting – Everyone
   - Complete/Update Iteration 1 Plan Document
   - Requirements
     - Begin Architecture for Iteration 1
   - Complete Initial Executive Summary
   - Update any changes in the pre-project notebook
   - Create Agenda for Week 7 meeting
   - Create Status Report 6 – Chase

3. Issues:
   - Snow storm causing an inability to meet in-person
     - Met at a determined time via Skype instead
   - Had trouble, due to weather, getting in touch with all members of team 5 P.O.T.

4. Scope:
   N/A

5 P.O.T. Status Report 18 February 2014

Name: "5POT Status Report – Week 6"
Purpose: Create Schedule for all iterations and complete the initial project notebook
Historical. Created each week.

1. Accomplished –
   - Planning Team Meeting - Everyone
     - Iteration Schedule
     - Requirements planned for iteration 1
     - Source control
     - Database discussion
   - Meeting through Skype on Saturday 2-15 at 1:00
   - Created tentative Iteration 1 schedule
   - Created Requirements Details for each requirements in Iteration 1
   - Create Status 6 Report - Chase
   - Create Agenda for Week 7 meeting – Chase

2. Planned – what is planned for next week
   - Iteration 1 Team Meeting – Everyone
     - Finalize Database schema
     - Plan next scheduled meeting
     - Plan additional tasks for the remainder of the week
   - Update Iteration 1 schedule
   - Requirements
     - Divvy up Requirements assignments responsibilities
     - Create storyboards for each requirement in Iteration 1
     - Create design URLs for the requirements.
   - Complete Initial Executive Summary
   - Update any changes in the Scope
   - Create Agenda for Week 8 meeting
   - Create Status Report 7 – Chase

3. Issues – new issues that have arisen this week, what has been done for issues on previous status reports
   - No issues

4. Scope – list any changes to scope
   - Architecture 1st Iteration - Design and implement the Database, create basic DB models, create ASP.NET MVC application, create WCF services, create interface models, and create relationships and connect all services
   - Architecture / Admin UI - Designing and implementing the administrative UI that all actors will use portions of
   - SA1 System Admin adds new employee to system
   - SA1 Alt1 System Admin removes employee from system
   - SA2 System Admin edits employee information

- SA3 System Admin changes permission level for employee
- SA4 System Admin fixes typo in questionnaire
- SE1 Staffing expert creates job title and description
- SE2 Staffing expert creates questions to be used in questionnaires
- SE3 Staffing expert creates questionnaire for a job
- SE4 Staffing expert edits a question from pool
- SE5 Staffing expert edits a questionnaire for a job
- NF1 Will be used by English speaking adults
- NF5 Assumed hardware used is Computer, Keyboard, and Mouse
- NF4 Disability accommodation is currently not needed

5 P.O.T. Status Report 25 February 2014

Name: "5POT Status Report – Week 7"
Purpose: Create Schedule for all iterations and complete the initial project notebook
Historical. Created each week.

1. Accomplished –
   - Design Team Meeting - Everyone
   - Meeting through Skype on Saturday 2-22
   - Sequence diagrams for in scope use cases for iteration 1
   - Began uncovering potential classes and methods
   - Finalized database design
   - Began building database

2. Planned – what is planned for next week
   - Construction team meeting
   - Begin dividing up architecture and code work
   - Tackle use cases
   - Create Status Report 8 – John

3. Issues – new issues that have arisen this week, what has been done for issues on previous status reports
   - Have questions for project manager

4. Scope – list any changes to scope
   - No changes

5 P.O.T. Status Report 4 March 2014

Name: "5POT Status Report – Week 8"
Purpose: Create Schedule for all iterations and complete the initial project notebook
Historical. Created each week.

1. Accomplished –
   - In-Class Team Meeting - Everyone
   - Finalized Models and detailed out the Controllers for each of the models.
   - Meeting through Skype on Friday 2/28
   - Finalized the controller details
   - Divided up the Controllers and assigned the creation of them to group members.
   - Database design was implemented on Azure

2. Planned – what is planned for next week continue working on the finer details of the implementation and work out the implementation of testing.
   - Continue dividing up architecture and code work and begin branching and merging code.
   - Create Status Report 9 – Jordan

3. Issues – new issues that have arisen this week, what has been done for issues on previous status reports.
   - None

4. Scope – list any changes to scope
   - No changes

5 P.O.T. Status Report 11 March 2014

Name: "5POT Status Report – Week 9"
Purpose: Communicate team's status to project manager
Historical. Created each week.

1. Accomplished –
   - Framework Setup Complete - Everyone
     - Database is working properly
     - Jobs, Users, and questions can be manipulated/added to/
   - Json parsing of survey questions demonstrations - Patrick
   - AIM Admin GUI and initial views complete- Jordan
   - Updated Defect Report – John
   - Updated report documents for portfolio – Everyone
   - Research on web-services nearing completion – Chase
   - Controllers communicating with services – John, Patrick, Josh
   - Create Agenda for Week 10 meeting – Jordan

2. Planned – what is planned for next week
   - Planning Team Meeting - Everyone
     - Create Requirements List
     - Identify Needs
     - Identify Risks
     - Identify Prioritizes
     - Create Estimates
     - Amend Scheduled Details
   - Test code - Everyone
   - Develop our presentation - Everyone
   - Walk through use cases - Everyone
   - Fix bugs with web-services - Chase
   - Create a basic login feature – Jordan
   - Update controllers to display correct data – Josh, Patrick, John
   - Add test data to database – Everyone

3. Issues – new issues that have arisen this week, what has been done for issues on previous status reports
   - Had trouble getting our services to connect to the database.
   - All previous issues have been resolved.

4. Scope – list any changes
   - No changes yet… but our timeline shows that possible cuts will happen next week.

5 P.O.T. Status Report 18 March 2014

Name: "5POT Status Report – Week 10"
Purpose: Create Agenda for 30-45 minute Iteration 1 Presentation, and get Iteration 1 prototype up and running.
Historical. Created each week.

1. Accomplished –
   - Two Team Meetings - Everyone
     - Everyone updated to the current project through Github
     - Create PowerPoint presentation slides for Iteration 1 Presentation
     - Fix 'edit' bug within the project
     - Fix formatting issues for pages within the project
   - Update Requirements, QA, and Architecture documents - Everyone
   - Compile and Format all updated documents into the project notebook – Chase
   - Create Status 10 Report - Josh
   - Create Agenda for next term meeting – Chase
2. Planned – what is planned for the next weeks
   - Spring Break!
   - Redefine scope and plan for Iteration 2
   - Requirements
     - Begin coding features and services for Iteration 2
   - Update any changes in the pre-project notebook
   - Create Agenda for Week 11/12 meeting
   - Create Status Report 11 – Chase
3. Issues – new issues that have arisen this week, what has been done for issues on previous status reports
   - Ran out of time in our initial meeting to get the whole presentation planned
     - Met via Skype at a later date before class
   - Had a lot of trouble merging local branches to Github and making those available for all team members
   - Trouble with Visual Studio "async" within the services
     - Found solution and issue has been resolved
4. Scope – List any changes to scope
   - We did not implement the Json logic for our questions, which caused quite a few use cases to be bumped out of scope, and in to the next iteration.
   - SA3 System Admin changes permission level for employee; permission levels were not implemented
   - All Question/Questionnaire Use cases moved out of scope:
     - SA4
     - SE2
     - SE3
     - SE4

5 P.O.T. Status Report 8 April 2014

Name: 5POT Team Status Report – Week 1
Purpose: Record accomplishments of previous, plans for upcoming weeks, issues, & scope
Type: Historical and is created each week.

1. Accomplished:
   - Iteration 2 Plan
   - Iteration 2 Scope
   - Detailed requirements for use cases in scope this iteration

2. Planned:
   - Work on design components including:
     - Sequence diagrams
     - Updating domain model
   - Talk about testing, something we did not cover last iteration

3. Issues: None

4. Scope:
   - Finish user login
   - Database validation of any information passed to it
   - Create questions and questionnaires
   - Create the ability to:
     - Fill out application
     - Apply for a job from a list of choices
     - Answer the questionnaire for the job
     - Be able to pass or fail the questionnaire for a job

5 P.O.T. Status Report 15 April 2014

Name: 5POT Team Status Report – Week 2
Purpose: Record accomplishments of previous, plans for upcoming weeks, issues, & scope
Type: Historical and is created each week.

1. Accomplished:
   - Sequence Diagrams created for each Requirements of Iteration 2
   - Story Boards created for Iteration 2
   - Started to create some unit tests that were left from Iteration 1
   - Worked through and started implementing JSON logic for Questions

2. Planned:
   - Work on coding components including:
     - Break up work for coding
     - Talk through the Sequence Diagrams
     - Brainstorm ideas for some coding solutions
   - Continue testing

3. Issues: None

4. Scope:
   - Finish user login
   - Database validation of any information passed to it
   - Create questions and questionnaires
   - Create the ability to:
     - Fill out application
     - Apply for a job from a list of choices
     - Answer the questionnaire for the job
     - Be able to pass or fail the questionnaire for a job

5 P.O.T. Status Report 22 April 2014

Name: 5POT Team Status Report – Week 4?
Purpose: Record accomplishments of previous, plans for upcoming weeks, issues, & scope
Type: Historical and is created each week.

1. 1. Accomplished:
   - UI creation for the Application process almost complete
   - HTML validation in the works, should be implemented by end of night
   - Added more unit tests
   - Started work on mock classes for testing.
   - UI for picking jobs and starting application has been implemented

2. Planned:
   - Work on coding components including:
     - Break up work for coding
     - Talk through the Sequence Diagrams
     - Brainstorm ideas for some coding solutions
   - Continue testing

3. Issues: Update for Visual Studio broke a library, worked with creator to fix(plus somebody just had to have a birthday!)

4. Scope:
   - Finish user login
   - Database validation of any information passed to it
   - Create questions and questionnaires
   - Create the ability to:
     - Fill out application
     - Apply for a job from a list of choices
     - Answer the questionnaire for the job
     - Be able to pass or fail the questionnaire for a job

5 P.O.T. Status Report 29 April 2014

Name: 5POT Team Status Report - Week 4
Purpose: Record accomplishments of previous weeks, plans for upcoming week, as well as any issues or scope changes.
Type: Historical and is created each week.

1. Accomplished:
   - Started testing
   - Added stores and regions
   - Began implementing job openings and being able to view and apply for them

2. Planned:
   - More in-depth testing
   - Finishing up the functionality that's being worked on.
     - Screening questions
     - Creating questionnaires
     - Searching for job openings

3. Issues:
   - Implementing screening questions the way we originally planned has been difficult.
   - Adding new functionality to existing services isn't working 100%
   - We've been out of touch with 2 team members, slowing our progress.

4. Scope:
   - No changes as of yet.

5 P.O.T. Status Report 6 May 2014

Name: 5POT Team Status Report - Week 5
Purpose: Record accomplishments of previous weeks, plans for upcoming week, as well as any issues or scope changes.
Type: Historical and is created each week.

1. Accomplished:
   - - use case AP1… home-screen that displays drop-down of jobs sorted by region. Includes a details button and a button to apply for that specific position.
     - Region functionality (Region based job view and search) has been completed and implemented into master.
   - Researched a strategy for handling questionnaires and applications using knockout.js.
   - Updated question objects so they work properly in terms of how they are to be parsed displayed and saved in the views.
   - Test cases were added to the master project and were running successfully.
   - Updated the project notebook documents that changed during this iteration.
   - Met online via Skype to discuss plans for notebook and the upcoming week.

2. Planned:
   - fixing up a few questionnaire issues with the knockout.js strategy
   - Finishing up the functionality that's being worked on.
     - database/permissions based log-in
     - Creating questionnaire and application views (GUI)
   - Collaborate on the iteration assessment after class.
   - Re-evaluate scope for next iteration, and make appropriate changes.

3. Issues:
   - Personal (family, health, work) issues have taken a toll on projected productivity.
     - Caused delays in getting certain functional requirements completed as originally planned.
   - Github, specifically the .ignore file not ignoring certain files.

4. Scope:
   - No changes as of yet… just moved a few features to the final iteration.

5 P.O.T. Status Report 13 May 2014

Name: 5POT Team Status Report - Week 5
Purpose: Record accomplishments of previous weeks, plans for upcoming week, as well as any issues or scope changes.
Type: Historical and is created each week.

1. Accomplished:
   - Updated Personal Info view.
     - Gave commonality among different views. Also gave default values for blank fields
   - Used knowout.js to fix issues w/ questionnaire
   - Updated display names to be more readable

2. Planned:
   - Finishing up the functionality that's being worked on.
     - Database/permissions based log-in
     - Creating questionnaire and application views (GUI)
   - Creating form validation for all parts of the project
   - Complete functionality for filling out/submitting an application

3. Issues:
   - Personal (health, vacation) have slowed productivity.
   - Github, specifically the .ignore file not ignoring certain files.

4. Scope:
   - Some Staffing Expert and Hiring Specialist specific use cases were pushed out of scope in order to make Iteration 3 more manageable.

5 P.O.T. Status Report 20 May 2014

Name: 5POT Team Status Report - Week 5
Purpose: Record accomplishments of previous weeks, plans for upcoming week, as well as any issues or scope changes.
Type: Historical and is created each week.

1. Accomplished:
   - .gitignore issue resolved
   - Can now push and pull without bin or obj folders interfering
   - Questions implemented
   - Began fixing bugs/defects
     - edit
   - login has basic functionality

2. Planned:
   - Implement questionnaire
   - Application functionality (Fill out/submit)
   - Requesting/Approving Job openings

3. Issues:
   - services break on update

4. Scope:
   - No changes

5 P.O.T. Status Report 27 May 2014

Name: 5POT Team Status Report - Week 5
Purpose: Record accomplishments of previous weeks, plans for upcoming week, as well as any issues or scope changes.
Type: Historical and is created each week.

1. Accomplished:
   - Implemented a final fix for the Services.
   - Login Permissions now accessible from all pages.
   - All members are now back and working!

2. Planned:
   - Wrap up current implementations
   - Start Test work.

3. Issues:
   - Major flaws in the service prevent doing work.

4. Scope:
   - Need to meet with team to make decisions on scope.

5 P.O.T. Status Report 3 June 2014

Name: 5POT Team Status Report - Week 4
Purpose: Record accomplishments of previous weeks, plans for upcoming week, as well as any issues or scope changes.
Type: Historical and is created each week.

1. Accomplished:
   - Implemented
     - Questions
     - Request/Delete/Approve job openings
     - Permissions filtering
     - Questionnaire
     - Application
   - Updated services
   - Updated notebook documents
   - Prepared for presentation

2. Planned:
   - Presentation

3. Issues:
   - Costing time
   - Create/Edit through the application has been problematic

4. Scope:
   - Time out features
   - Create questionnaire through the site/app
   - HTML validation

5 P.O.T. Status Report 10 June 2014

Name: 5POT Team Status Report - Week 4
Purpose: Record accomplishments of previous weeks, plans for upcoming week, as well as any issues or scope changes.
Type: Historical and is created each week.

1. Accomplished:
   - Reworked the Services again and got them partially updated (50%)
   - Implemented a GUI technique for stepping through application and questionnaires
   - Added some permissions filtering to the admin site
   - Tested out how the implementation for questionnaires and applications would work.
   - Fixed bugs on already created views.

2. Planned:
   - Finish Implementing:
   - Questions
   - Request/Delete/Approve job openings
   - Permissions filtering
   - Finish Implementing:
   - Questionnaire
   - Application
   - Updated services
   - Updated notebook documents
   - Prepared for presentation

3. Issues:
   - Services still needed to be finalized… and blocked certain features from being implemented or tested
   - It's been increasingly difficult to get everyone to meet at the same time with finals approaching.

4. Scope:
   - Cleaning up the application GUI
   - Create questions that work for multiple questionnaires

Requirements

A.I.M. Requirements Document

Name: A.I.M. Requirements
Purpose: To create an automated kiosk for AES where potential applicants can apply for jobs at.
Living: Created during the pre-project, updated during requirements phase of iterations.

Overview:

The goal is to create a kiosk where an individual can apply for AES retail positions at any of the surrounding metropolitan AES stores. The kiosk will allow potential applicants to search for job openings in a selected region. The applicant will then be able to select a job to apply for and be presented with a questionnaire pertaining to the position they chose. Should an applicant pass the questionnaire screening they will then be asked to enter in personal and contact information, as well as be given instructions on how to proceed through the next phase of the application process. Applicants who do not pass the questionnaire screening will have any information that was stored discarded.

Administrative positions will also interact with the system. Staffing Experts will access the system to create job titles and descriptions. They will also create and add questions to both kiosk questionnaires and phone interviews. Hiring managers may request job openings through the system that are then reviewed and either approved or denied by their Store Manager for that store. Hiring Specialists will conduct phone interviews with applicants who have passed the questionnaire screening and record questions and responses in the system. Should an applicant pass the phone interview they will be directed to a Hiring Manager for an in-person interview.

The Hiring Manager will be able keep notes about the applicant's interview in the system and decide whether they should no longer remain in the applicant pool, keep them in the pool, or request to hire the applicant. Upon request, Hiring Specialists will conduct a background and reference check on the applicant and change the applicant's status to either hired or denied due to a concern from the checks. Finally, an administrator will need to manage accounts and permissions for each of these positions.

Actors:

Applicant

- Completes questionnaire
- Submits applications after passing questionnaire

System Admin

- Add users
- Manage permissions for system

- Perform any task other managers and specialists can

Store manager

- Approves positions
- Creates openings for his own positions (i.e. management)

Hiring Manager

- Requests positions to open
- Conducts in person interviews
    - Keep in hiring pool
    - Remove from pool
    - Recommend for hiring
- Sees applications for the positions they oversee in that store

Types and what they hires:

Operations Manager hires:

- Store clerks
- Custodians
- Sales clerks

Dept. Manager hires:

- Sales associate
- Purchaser
- Product Specialist

Staffing expert

- Creates job titles and descriptions
- Creates questions for questionnaires and phone interviews
- Adds questions to questionnaires and phone interviews

Hiring specialist

- Does phone interviews
- Accesses application info
- Record questions asked
- Approve/disapprove applicant (with reason)
- Does background checks and checks references (after hiring manager recommendation)

Database

- Stores applications and their status
- Stores jobs and descriptions
- Stores job questionnaires
- Stores available job openings
- Stores job openings to be approved

Uses Cases:

Definitions and Explanations:

Administrative Home Page view includes links of the following views; User Login List View, User List View, Job List View, Question List View .....

Staffing Expert home page includes default job list view...

Job List View includes list of all jobs with options to create, edit, and delete.

Job Detail view includes job name, description, and Questionnaire list of question assigned to the job...

Job Openings View includes selecting the region and store the user wishes to view job openings in. Upon selection, a list of currently approved and pending approval openings displays.

Question List view includes list of questions in the question pool with options to create, edit, and delete.

Edit Question View includes fields for Title, format of question, format of answer, text field

Create Questionnaire view includes Questionnaire title, plus the Edit Questionnaire view…

Edit Questionnaire View includes list of questions, with questions assigned to current questionnaire highlighted…

Employee details include username, password, and position in the company.

User Login List View includes all system usernames and passwords with options to create, delete, and edit.

User List View lists all users in the system with options to create, delete, and edit.

Valid field details include non-null fields, english alphanumeric characters, dashes (possibly more to come).

SA1 - System Admin adds new employee to system

Overview: System Administrator creates a new user account for an employee and adds user details for the employee.

Description: The System Administrator (SA) will navigate to the admin home page. They will then select an option to view the user login list and choose the option to add a new user to the system. The SA will enter in the various details for the new user (who is an employee working for the company) and select the option to submit the new user to the database for storage. Possible alternate flows for this use case include a user account already existing or the SA entering invalid field data when creating a new user.

Main Flow:

1. System Administrator navigates to the admin home page
2. System displays administrative home page view
3. System Administrator selects option to open user login list view
4. System displays user login list view
5. System Administrator chooses option to add new user account.
6. System displays create new user view.
7. System Administrator enters new user details, permissions, and selects submit option.
8. Administrative Service validates data and updates database.
9. System displays confirmation.
10. System Administrator acknowledges confirmation.
11. System returns to admin home page.

Alternate 8A – Account already exists

1. Administrative Service sends invalid exception
2. System displays account already exists message.
3. System Administrator acknowledges message
4. System resumes at Main flow step 4.

Alternate 8B – Invalid field data

1. Administrative Service sends invalid exception
2. System displays and highlights invalid fields message.
3. System Administrator acknowledges message
4. System resumes at Main flow step 4.

SA1 Alt1 System Admin removes user login accounts from system

Overview: System Administrator deletes a user login preexisting account for an employee.

Description: The System Administrator (SA) will navigate to the admin home page and then select the option to view the user login list. They will select the option to delete a chosen user's login account. The System will ask the SA for confirmation of this action, after which the System will delete the login account for the chosen user from the database.

Main Flow:

1. System Administrator navigates to the admin home page
2. System displays administrative home page view
3. System Administrator selects option to open user login list view
4. System displays user login list view
5. System Administrator selects option to delete the identified user login

6. System displays message to confirm System Administrator wants to delete user login account.
7. System Administrator selects confirm option
8. System submits action to delete user account to database and displays confirmation message
9. System Administrator acknowledges confirmation
10. System displays user login list view

SA2 System Admin edits employee information

Overview: System Administrator (SA) opens employee information page from list, edits fields, and submits changes to database.

Description: The SA will navigate to the admin home page and select the option to view the user login list. The SA will then select the option to edit a specific user. After editing the necessary fields the SA will select the option to save the changes. After confirming the changes with the System, the edited details will be stored on the database.

Main Flow:

1. System Administrator navigates to the admin home page
2. System displays administrative home page view
3. System Administrator selects option to open user login list view
4. System displays user list view
5. System Administrator selects option to edit the identified user
6. System displays user details view
7. System Administrator edits fields that need revision and selects save option
8. System displays message to confirm editing user details
9. System Administrator selects confirm option
10. System validates data, submits to database, and displays confirmation
11. System Administrator acknowledges confirmation
12. System displays user details view

Alternate 10A - Information field(s) contain(s) incorrect data

1. System displays user details view, highlight invalid fields
2. Flow resumes at Main Flow 7

SA3 System Admin changes permission level for employee

SA opens employee information page from list, edits permission field and submits changes to database.

Permission level is a hidden field viewable only by the system administrator and can be viewed in employee details. The main flow is the same as SA2.

SA4 System Admin fixes typo in question

SA opens question from list, edits question fields, submits changes to database.

Main Flow:

1. System Administrator navigates to the admin home page
2. System displays administrative home page view
3. System Administrator selects option to open question list view
4. System displays question list view
5. System Administrator selects option to edit the identified question
6. System displays question details view
7. System Administrator edits fields that need revision and selects save option
8. System displays message to confirm editing question details
9. System Administrator selects confirm option
10. System validates data, submits to database, and displays confirmation
11. System Administrator acknowledges confirmation
12. System displays question details view

Alternate 10A - Information field(s) contain(s) incorrect data

1. System displays user details view, highlight invalid fields
2. Flow resumes at Main Flow 7

AP1 Applicant Views and Applies for Jobs

Here an applicant will sit down at a kiosk and begin filling out a job questionnaire. Should the applicant pass the questionnaire screening, they will be asked to created a username and password login as well as be given instructions for the phone interview process.

Main flow:

1. Applicant goes to kiosk homepage
2. App server lists jobs for metropolitan area
3. Applicant selects job to apply for
4. App server loads and goes to questionnaire view
5. Applicant inputs answers to questionnaire

6. Applicant submits questionnaire
7. App server evaluates applicant's submission
8. App server approves submission
    a. Alternate: App server denies submission
9. App server displays message approving applicant
    a. Alternate: App server discards submission
10. App server loads login/create account view
    a. Alternate: App server returns to homepage. End of Alternate flow
11. Applicant selects create account
    a. Alternate: Applicant selects log into account
12. App server loads create account view
13. Applicant inputs needed information
14. Applicant creates account
15. App server validates with database
16. App server creates new account in database
    a. Alternate: Validation fails
17. App server sends questionnaire in account to database
18. App server loads application view
19. Applicant inputs information into fields
20. Applicant submits application
21. App server sends application information in account to database
22. App server displays contact message for phone interview
23. App server sets applicant status to approve 1.

Alternate 8A App Server Denies Submission

AP2 Applicant creates an user account with AES

The Applicant creates an user account with AES after selecting a job to apply for and has successfully completed the associated questionnaire to access and create an application.

Use Case AP3 Applicant completes questionnaire is performed prior to AP2's Main Flow with Applicants inputs meeting questionnaire's correct answers.

Create User Account View includes:

- message to Applicant stating that the Applicant meets the minimum requirements for the job they selected to apply to and asks them to please create/login an AES account to continue that application process.
- Input fields for firstName, middleName, lastName, email, socialSecurityNumber, userName and password.
- Options for existing account, submit new account, and cancel.

Questionnaire data includes the following fields: jobId, questionnaireId, and applicantAnswers

Application Index View includes: input fields for application…

Login View includes: Input fields for userName and password and option for Login, return to Create User, or cancel.

Validation rules will return message to Applicant about validation rule broken and asks them to reenter information and returns to Main Flow 2:

1. Applicant inputs meets database rules for each element.
2. New user is not matching an existing user.
3. Existing user matches userName and password.

Main Flow:

1. Client Web system displays Create User Account View.
2. Applicant fills out input fields and selects submit new account option.
3. Client Web system pushes input fields and questionnaire data to Authentication Service.
4. Authentication Service send a request to AIM Database to Create new user with received input fields and questionnaire data.
5. AIM Database creates new User and associated Applicant and returns user to Authentication Service.
6. Authentication Service returns user to Client Web system.
7. Client Web system displays Application Index View.

Alternate 2A - Applicant has an existing account.

1. Applicant selects existing account option.
2. Client Web system displays Login View.
3. Applicant fills out input fields and selects login option.
4. Client Web system pushes input fields and questionnaire data to Authentication Service.
5. Authentication Service send a request to AIM Database to Pull user and Update received questionnaire data.
6. AIM Database updates user and returns user to Authentication Service.
7. return to Main Flow 6.

Alternate - Applicant selects cancel option.

1. Applicant selects cancel option.
2. Client Web system displays message to verify cancel and option to verify or cancel.
3. Applicant selects verify option.
4. Client Web system returns to Client Web Home page.

AP3 Applicant completes questionnaire

After selecting the jobs the applicant wants to apply for, they are directed to the questionnaire view. The applicant will answer each question and then select a submit button which will inform the applicant if they pass or fail the questionnaire. If the applicant passes they will create a login or application.

Definitions:

Question View / Options -

1. One per page…
    a. each page has a single question with a "back" and "next" button. The current progress (ie: Question 1 of 10) is displayed on each page. The last question page will have the submit questionnaire button. There is also a cancel button on each question page so the applicant can quit at any time… this button will return the view to the kiosk home page.
2. All in one view (scrolling)…
    a. All questions will be on a single view that will require the applicant to scroll should they not fit on a single page. The submit button will appear at the bottom once all questions have been answered. There is also a cancel button (at the top and bottom of the page?) that will return the view to the kiosk home page.

Timeout: One minute of inactivity will cause the kiosk to reset to home page.

Main Flow:

1. Applicant arrives at the kiosk and "wakes" it up.
2. AIM displays current local jobs.
3. Applicant browses jobs and selects to apply for one.
4. AIM displays the questionnaire.
    a. AIM requests questionnaire for selected job from the questionnaire service.
    b. Questionnaire service returns the requested questionnaire to AIM..
5. Applicant fills out answers to questionnaire questions.
   see "Question View / Options"

1. Applicant selects to submit the questionnaire.
2. AIM determines the applicant has passed and loads the SUCCESS view.
    a. AIM requests questionnaire service to check the answers the applicant submitted against the correct answers and ALL answers pass.
    b. Questionnaire service returns 'SUCCESS'
    c. AIM displays the "Questionnaire Passed View"

3. Applicant selects option to continue to create account page, go to AP2.
4. AIM retains applicant answers to be saved in applicant files on server after the AP2 is completed.

Alternate 5a.

1. Applicant leaves Kiosk and system times out.

Alternate 5b.

1. Applicant clicks submit or next before completing one or more question(s).
2. AIM displays error message indicating question(s) must be answered before moving on.

Alternate 5b.

1. Applicant clicks cancel button.
2. AIM displays an 'are you sure' confirmation pop-up / view.
3. Applicant confirms.
4. AIM displays the kiosk home page.

AP4 Applicant fills out and submits applications after passing questionnaire

After successfully passing a questionnaire screening and creating a user account, the applicant fills out an application to be stored away and used during the rest of the application process. It is assumed the applicant is logged into their user account (See AP2).

Definitions:

Application: Contains the applicant's name, vital, and contact information, references, and education and job history. All of the information is required to be filled out with the exception of job history.

New Application View: A view showing all fields of a an application that are to be filled out the applicant.

User Application List View: Lists all applications the applicant has filled out and saved.

Main Flow:

1. System displays option to enter in a new application or use an existing application for the completed questionnaire.
2. Applicant selects to enter in a new application.
3. System displays new application view.
4. Applicant enters in all information for an application.
5. Applicant selects option to save application.
6. System pushes contents of the input fields to the Application Service for validation.
7. Application Service validates input fields
8. Application Service saves application to database.
9. System displays confirmation the application has been saved.
10. Applicant acknowledges confirmation.
11. System displays User Application List View.

Alternate 2A - Applicant selects option to use an existing application.

1. Application service gets all applications saved for user account.
2. System displays all applications retrieved by Application Service.
3. Applicant selects the application to use.
4. Applicant confirms the selected application to be used.
5. Application saves selection.
6. Return to Main Flow 9.

Alternate 7A - Input fields contain invalid data or a required field(s) is empty

1. System displays New Application with the information the applicant previously entered and also displays which fields are incorrect.
2. Applicant corrects the appropriate fields.
3. Return to Main Flow 5.

AP4-Alt Applicant does not pass questionnaire and applies for a different job

The applicant takes the questionnaire and incorrectly answers one or more questions. After being informed by the system that they have not passed the questionnaire, the user is sent to the job list page again and selects a different job to apply for.

Failure Page: This page indicates that the user did not answer all of the questions correctly and displays an option to return to the Applicant menu(job list) to either reapply or apply for a different job. Has a time-out of 1 minute which will automatically redirect to the Main job listing page.

Main Flow:

1. Applicant arrives at the kiosk and "wakes" it up.
2. Client Web system displays current local jobs.
3. Applicant browses jobs and selects to apply for one.
4. Client Web system requests questionnaire for selected job from the questionnaire service.
5. Questionnaire service returns the requested questionnaire to Client Web system.
6. Client Web System displays the questionnaire.
7. Applicant fills out answers to questionnaire questions.
8. Applicant selects to submit the questionnaire.
9. Client Web system checks the answers the applicant submitted against the correct answers and finds mismatches.
10. Client Web system displays displays the failure page.
11. Applicant selects option to return to main page, go to AP3.
    a. ALT 11a. Applicant leaves Kiosk and system times out.

AP5 Applicant calls HS for a phone interview (See HS4

After the applicant completes, and passes the questionnaire and submits the application (AP3 & AP4), the applicant is provided a phone number to call and participate in an over-the-phone interview with a hiring specialist.

Main Flow:

1. (AP4) Client Web system pushes application field data to Auth Service
2. (AP4) Application Service sends push request to AIM Database to insert application data
3. Application Service sends update request to AIM Database to update applicant application status to "phone-interview ready"
4. Client Web system displays submittal confirmation for application
5. Applicant acknowledges submittal confirmation
6. Client Web system displays instructions and phone number for the phone interview
7. Applicant locates store phone and calls the Hiring Specialist for the phone interview

SE1 Staffing expert creates job title and description

Staffing expert will create a brand new job title. They will then fill out the necessary description for that job.

Main Flow:

1. Staffing Expert navigates to the Staffing Expert home page
2. System displays Staffing Expert home page view
3. Staffing Expert chooses option to add new job.
4. System displays create new job view.
5. Staffing Expert enters job details and selects submit option.
6. Administrative Service validates data and updates database.
7. System displays confirmation.
8. Staffing Expert acknowledges confirmation.
9. System returns to Staffing Expert home page.

Alternate 6A – Job already exists

1. Administrative Service sends invalid exception
2. System displays job already exists message.
3. Staffing Expert acknowledges message
4. System resumes at Main flow step 4.

Alternate 6B – Invalid field data

1. Administrative Service sends invalid exception
2. System displays and highlights invalid fields message.
3. System Expert acknowledges message
4. System resumes at Main flow step 4.

SE2 Staffing expert creates questions for questionnaires

The Staffing Expert creates a new question and adds it to the question pool.

Main Flow:

1. Staffing Expert navigates to the Staffing Expert home page
2. System displays Staffing Expert home page view
3. Staffing Expert selects option to navigate to Question List view
4. System displays Question List view
5. Staffing Expert chooses option to add new question.
6. System displays create new question view.
7. Staffing Expert enters question details and selects submit option.
8. Administrative Service validates data and updates database.
9. System displays confirmation.

10. Staffing Expert acknowledges confirmation.
11. System returns to Staffing Expert home page.

Alternate 6A – Question already exists

1. Administrative Service sends invalid exception
2. System displays question already exists message.
3. Staffing Expert acknowledges message
4. System resumes at Main flow step 4.

Alternate 6B – Invalid field data

1. Administrative Service sends invalid exception
2. System displays and highlights invalid fields message.
3. System Expert acknowledges message
4. System resumes at Main flow step 4.

SE3 Staffing expert creates questionnaire for a job

The Staffing Expert selects questions from the pool. These selected questions

will compose a questionnaire that will be linked to a particular job.

Main Flow:

1. Staffing Expert navigates to the Staffing Expert home page
2. System displays Staffing Expert home page view
3. Staffing Expert selects a job from list.
4. System displays job detail view.
5. Staffing Expert selects option to create questionnaire for job.
6. System displays Create Questionnaire view
7. Staffing Expert selects existing questions to add to questionnaire and selects option to add
8. Administrative Service validates data and updates database.
9. System displays confirmation.
10. Staffing Expert acknowledges confirmation.
11. System returns to Job Detail page.

Alternate 7A - Staffing Expert selects option to create new question for questionnaire

1. System displays create new question view.
2. Staffing Expert enters question details and selects submit option.
3. Administrative Service validates data and updates database.
4. System displays confirmation.
5. Staffing Expert acknowledges confirmation.
6. System resumes at Main Flow step 7

Alternate 7A Alternate 3AA – Question already exists

1. Administrative Service sends invalid exception
2. System displays question already exists message.
3. Staffing Expert acknowledges message
4. System resumes at Alternate 7A step 2.

Alternate 7A Alternate 3BB – Invalid field data

1. Administrative Service sends invalid exception
2. System displays and highlights invalid fields message.
3. System Expert acknowledges message
4. System resumes at Alternate 7A step 2

.

SE4 Staffing expert edits a question from pool

The Staffing Expert selects a question currently in the pool and edits it.

Main Flow:

1. Staffing Expert navigates to the Staffing Expert home page
2. System displays Staffing Expert home page view
3. Staffing Expert selects option to navigate to Question List view
4. System displays Question List view
5. Staffing Expert chooses option to navigate to Edit Question View.
6. System displays Edit Question View.
7. Staffing Expert edits question details and selects submit option.
8. Administrative Service validates data and updates database.
9. System displays confirmation.
10. Staffing Expert acknowledges confirmation.
11. System returns to Question List view.

Alternate 8A - Information field(s) contain(s) incorrect data

1. System displays Edit Question view, highlight invalid fields
2. Flow resumes at Main Flow step 7

SE5 Staffing expert edits a questionnaire for a job

The Staffing Expert selects a questionnaire currently linked to a job and edits it.

Main Flow:

1. Staffing Expert navigates to the Staffing Expert home page

2. System displays Staffing Expert home page view
3. Staffing Expert selects a job from list.
4. System displays job detail view.
5. Staffing Expert selects option to Edit questionnaire for job.
6. System displays Edit Questionnaire view
7. Staffing Expert selects existing questions to add to questionnaire and selects option to add
8. Administrative Service validates data and updates database.
9. System displays confirmation.
10. Staffing Expert acknowledges confirmation.
11. System returns to Job Detail page.

Alternate 7A - Staffing Expert selects option to remove question from questionnaire

1. System displays message to confirm deletion.
2. Staffing Expert confirms deletion
3. Administrative Service updates database.
4. System displays confirmation.
5. Staffing Expert acknowledges confirmation.
6. System resumes at Main Flow step 6

Alternate 7B – Added question is already in questionnaire.

1. Administrative Service sends invalid exception.
2. System displays question already in questionnaire message.
3. Staffing Expert acknowledges message.
4. System resumes at Main Flow step 6.

HS1 Hiring Specialist conducts phone interview

Here, a Hiring Specialist will view an applicant's information while conducting a phone

interview with them. The Hiring Specialist will be prompted by the system to ask the applicant questions and the Hiring Specialist will record their answers. Finally, the Hiring Specialist will either approve the applicant for an in-person interview or deny them.

Main Flow:

1. Hiring Specialist (HS) goes to admin login page
2. App server loads login page
3. HS enters login info
4. HS submits login
5. App server validates login information
6. App server pulls user info from database

a. Alternate: Validation fails
7. App server pulls applicants with completed Approval 1 status from database
    a. Alternate: App server displays login page. End Alternate flow.
8. App server loads HS view
9. HS selects applicant from list
    a. Alternate: HS searches for an applicant
10. App server pulls applicant's account information from database
    a. Alternate: App server pulls matching applicant account information from
        i. database. End of Alternate Flow
11. App server loads application interview view with notes field
12. HS fills in notes from interview
13. HS applicant status: Pass
    a. Alternate: HS applicant status: No Pass
14. HS changes applicant approval status from 1 to approval 2
    a. Alternate: HS changes applicant status to Not Approved
15. App server updates applicant data in database
16. App server loads HS view, End of Main Flow

HM1 Hiring Manager Requests A Job

The Hiring Manager accesses the system and selects job openings they would like to create.

Job Openings View:

Main Flow:

1. Hiring Manager (HM) navigates to Admin login page
2. App server loads login page
3. HM enters login info
4. HM submits login
5. App server validates login information
6. App server Displays Admin homepage
7. HM selects option to view current job openings
8. App Server displays Job Openings View
9. HM selects the region they are in
10. App server displays all stores in the selected region
11. HM selects the store they wish to view job openings in
12. App Server displays all job openings for the selected store
13. HM selects option to create a new job opening
14. App Server displays a create view for a new job opening
15. HM enters relevant information
16. App server sends the new, unapproved, job opening to database

End Main Flow

SM1 Store Manager Approves Job

Store Manager accesses the system, views opening requests made by the hiring manager and either approves or denies them.

The first steps of this use case are identical to HM1, only using the store managers credentials.

Resume flow at HM1 step 12.

1. Store Manager (SM) selects a currently unapproved job opening from list
2. App server gets information for selected opening and displays
3. Store Manager selects option to approve the opening
4. App server updates status of job opening

End Main Flow

HM2 Hiring Manager views application, conducts interview, and changes pool status

HM views an application while conducting an interview and records notes in the system

about the applicant. Finally, he requests to hire the applicant.

SM2 Store manager opens job

Identical to HM1, but instead uses the Store Managers credentials for login

HS2 Hiring Specialist views list of applicants for a specific job listing

The Hiring specialist selects a specific job listing OR a list of all applicants currently awaiting a phone interview.The list can be arranged by name, job listing or time application was submitted.

HS3 Hiring Specialist views an applicant's information

The Hiring Specialist can view the applicants resume, survey questions, which jobs they applied for (these may be different windows).

HS4 Hiring Specialist checks applicant references

The Hiring Specialist can view the applicants resume where the references are listed.

They can then call or check the resources validity and make remarks in the application as to whether they have been confirmed and if they are positive or negative reviews.

HS5 Hiring Specialist edits an applicant's interview

The Hiring Specialist can edit the information they added to an applicant information if mistakes were made. They might even be able to toggle whether an applicant is approved or denied.

Non-functional Requirements:

1. Will be used by English speaking adults
2. Can expect 1000 or more users accessing the system
3. Allowed downtime is when no stores are open
4. Disability accommodation is currently not needed
5. Assumed hardware used is Computer, Keyboard, and Mouse

Risk/Need/Estimate Analysis

ID#	Use Case #	Description	Need	Risk	Time Est. (In Days)	Additional Notes
50	Architecture 1st Iteration	Design and implement the Database, create basic DB models, create ASP.NET MVC application, create WCF services, create interface models, and create relationships and connect all services	High	High	15	
1	A1	A1 - Applicant chooses a job to apply for	High	High	15	
51	Architecture / Admin UI	Designing and implementing the administrative UI that all actors will use portions of	High	High	2.5	
34	HR3	Hiring Manager reviews applications	High	Med	7	
47	NF3	Allowed downtime is when no stores are open	High	Med	5	
3	A3	A3 - Applicant completes questionnaire	High	Med	4	
4	A4	A4 - Applicant fills out and submits applications after passing questionnaire	High	Med	4	
29	HR2 MF	Hiring Manager conducts in person interview and recommends to hire	High	Med	4	
40	HS1	Hiring Specialist views list of applicants for a specific job listing	High	Med	4	
41	HS2	Hiring Specialist views an applicant's information	High	Med	4	
42	HS3	Hiring Specialist checks applicant references	High	Med	4	
43	HS4	Hiring Specialist conducts a phone interview	High	Med	4	
44	HS5	Hiring Specialist edits an applicant's interview	High	Med	4	
21	SM1	Store Manager accesses position list and approves a position.	High	Med	3	

23	SM2	Store Manager creates openings for a management position.	High	Med	3	
37	SE3	Staffing expert creates questionnaire for a job	High	Med	3	
27	HR1 MF	Hiring Manager requests positions to open	High	Low	2	
2	A2	A2 - Applicant creates an user account with AES	High	Low	1.5	
16	SA1	System Admin adds new employee to system	High	Low	1.5	
18	SA2	System Admin edits employee information	High	Low	1.5	
35	SE1	Staffing expert creates job title and description	High	Low	1.5	
36	SE2	Staffing expert creates questions to be used in questionnaires	High	Low	1.5	
5	A4 Alt1	Applicant does not pass questionnaire and applies for a different job	High	Low	1	
7	A5	Applicant calls HS for a phone interview (See HS4)	High	Low	0.5	
9	A6	Applicant has in-person interview after passing the phone interview	High	Low	0.5	
11	A7	Applicant gets recommended for Hire after in-person interview	High	Low	0.5	
12	A7 Alt1	Applicant does not pass in-person interview and is removed from hiring pool	High	Low	0.5	
14	A9	Applicant is hired after background and reference checks are passed	High	Low	0.5	Clarify
15	A9 Alt1	Applicant does not pass background and reference checks, and is removed from hiring pool	High	Low	0.5	
52	HS6	Hiring Specialist removes applicant after failing background/reference checks	High	Low	0.5	
45	NF1	Will be used by English speaking adults	High	Low	0	
49	NF5	Assumed hardware used is Computer, Keyboard, and Mouse	High	Low	0	
26	SM4	Store Manager acts as Hiring Manager, with Hiring Manager permissions, for interviews for management positions.	Med	Med	4	
19	SA3	System Admin changes permission level for employee	Med	Low	1.5	
8	A5 Alt1	Line is busy and Applicant is asked to call back later	Med	Low	0.5	
20	SA4	System Admin fixes typo in questionnaire	Low	Low	2	
28	HR1 Alt1	Hiring Manager requests positions to close	Low	Low	2	
32	HR2 Alt 3	Hiring Manager calls to schedule and is unable to contact applicant	Low	Low	2	
33	HR2 Alt4	Applicant declines the position when called	Low	Low	2	
38	SE4	Staffing expert edits a question from pool	Low	Low	2	
17	SA1 Alt1	System Admin removes employee from system	Low	Low	1.5	
25	SM3	Store Manager accesses position list and reviews position details.	Low	Low	1.5	

6	A4 Alt2	Applicant does not pass questionnaire and doesn't apply for other jobs	Low	Low	1	
10	A6 Alt1	Applicant does not pass phone interview and is removed from hiring pool	Low	Low	0.5	
13	A8	Applicant is subject to background and reference checks	Low	Low	0.5	Clarify
48	NF4	Disability accommodation is currently not needed	Low	Low	0	
46	NF2	Can expect 1000 or more users accessing the system	Med	High	7	
30	HR2 Alt1	Hiring Manager conducts in person interview and declines	Low	Med	4	
31	HR2 Alt2	Hiring Manager conducts in person interview and places in pool	Low	Med	4	
39	SE5	Staffing expert edits a questionnaire for a job	Low	Med	4	
22	SM1 Alt1	Store Manager accesses position list and declines a position.	Low	Med	3	
24	SM2 Alt1	Store Manager removes openings for a management position.	Low	Med	3	

TABLE 5 – A.I.M. RISK/NEED/ESTIMATE ANALYSIS

A.I.M. Domain Model

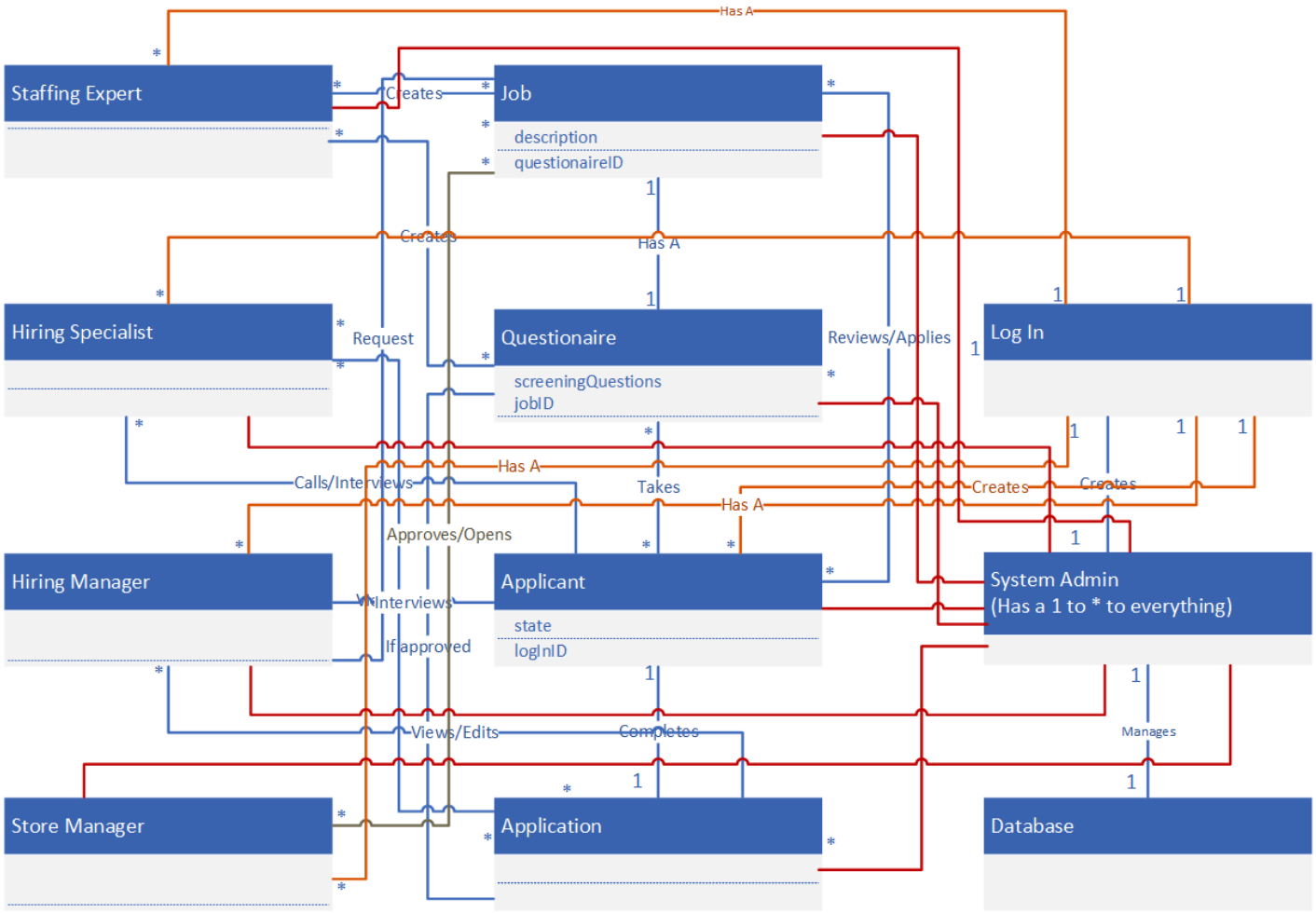


TABLE 6 - A.I.M. DOMAIN MODEL

A.I.M. Storyboards

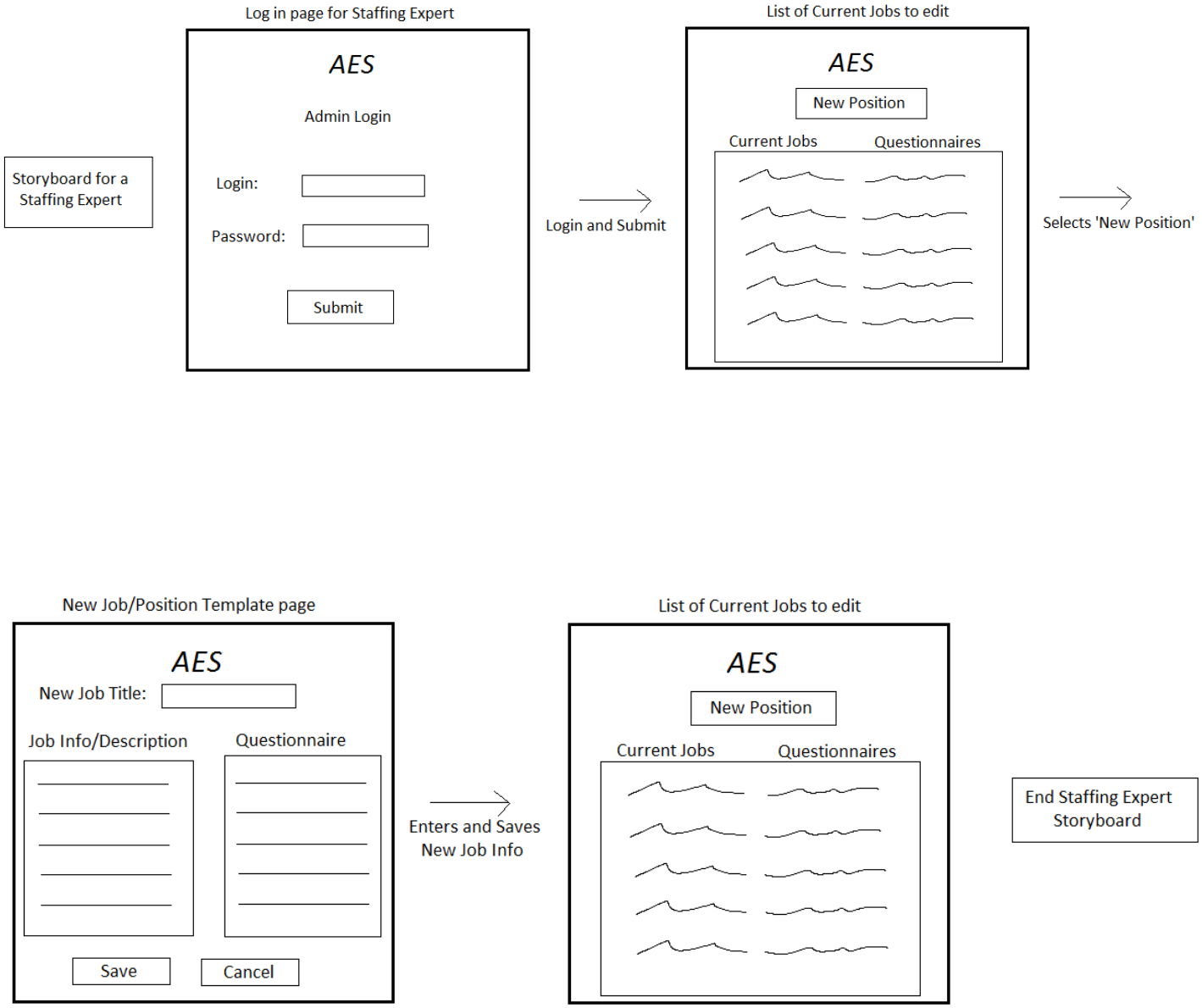


TABLE 7 - A.I.M. SE1 STORYBOARD

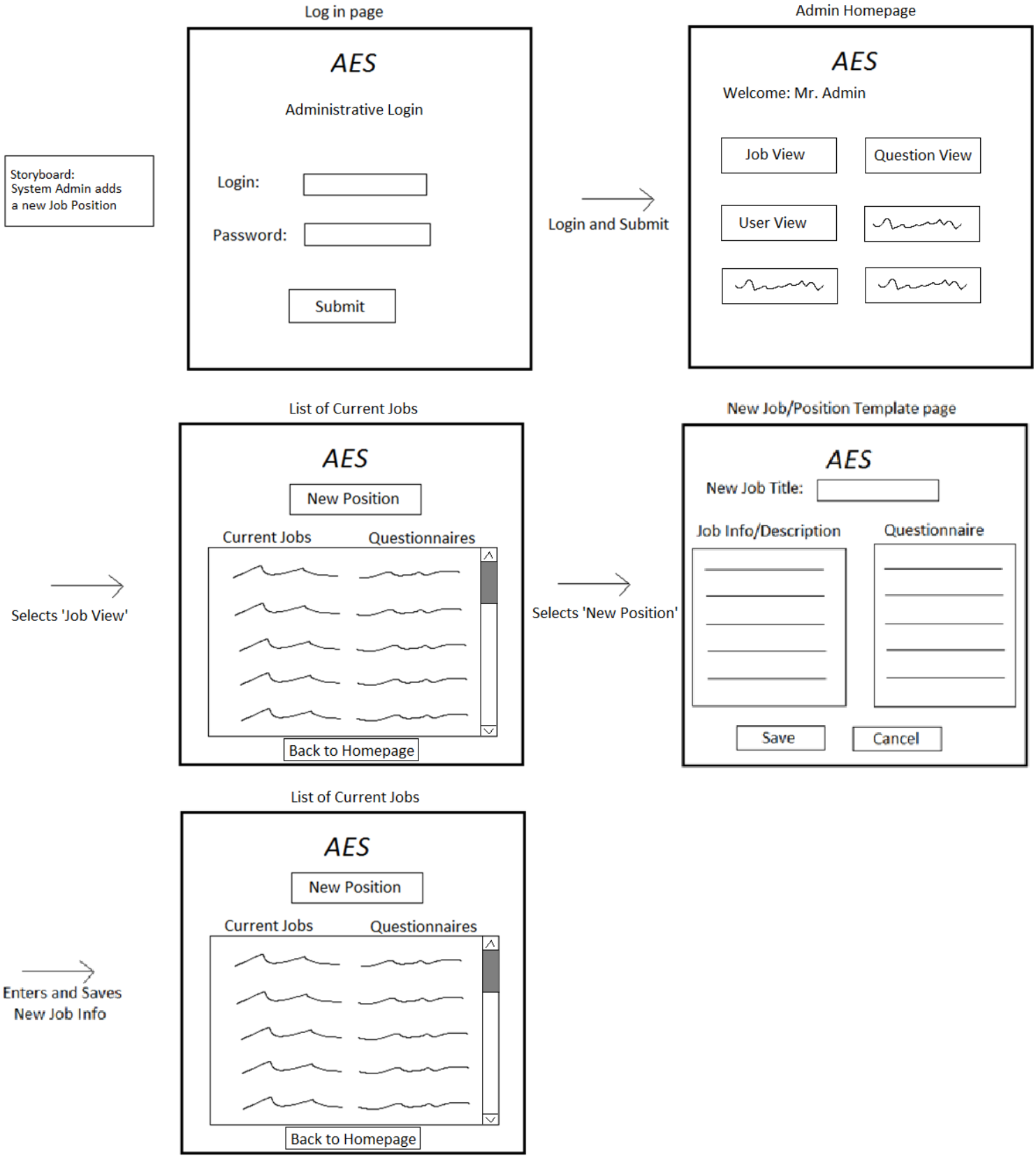


TABLE 8 - SA ADDS JOB STORYBOARD

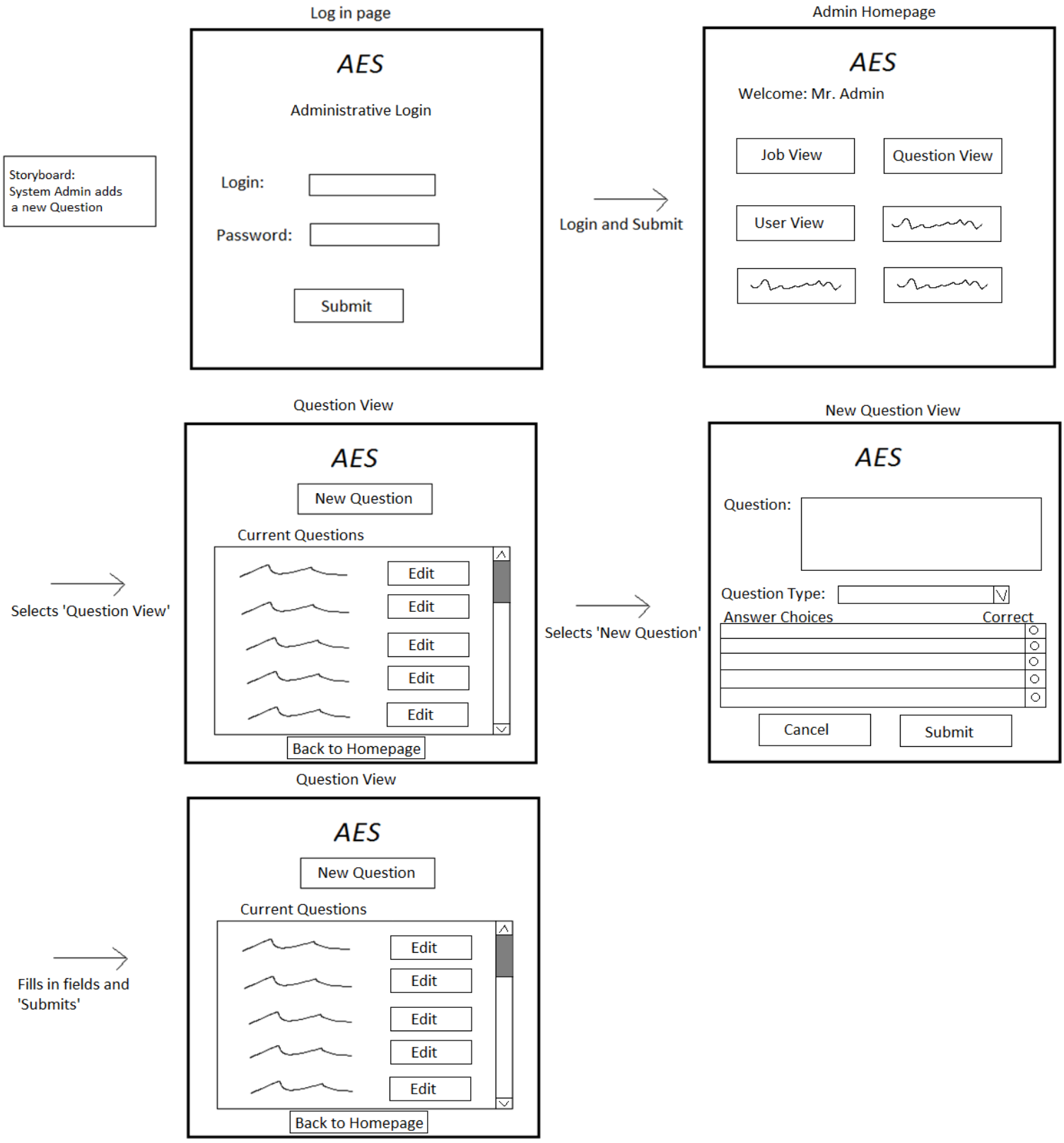


TABLE 9 - SA ADDS QUESTION STORYBOARD

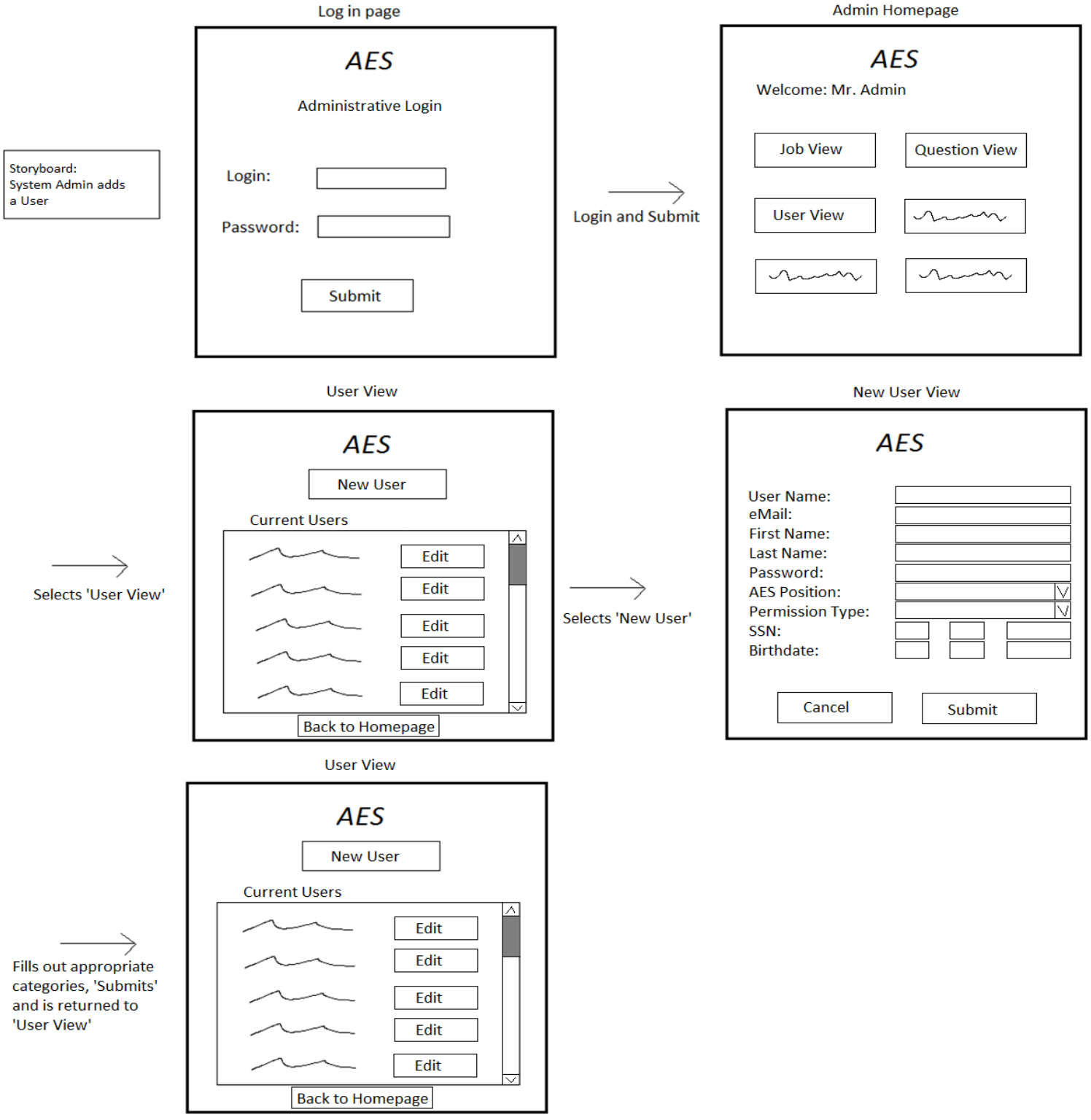


TABLE 10 - SA ADDS USER STORYBOARD

TABLE 11 - AP1 STORYBOARD

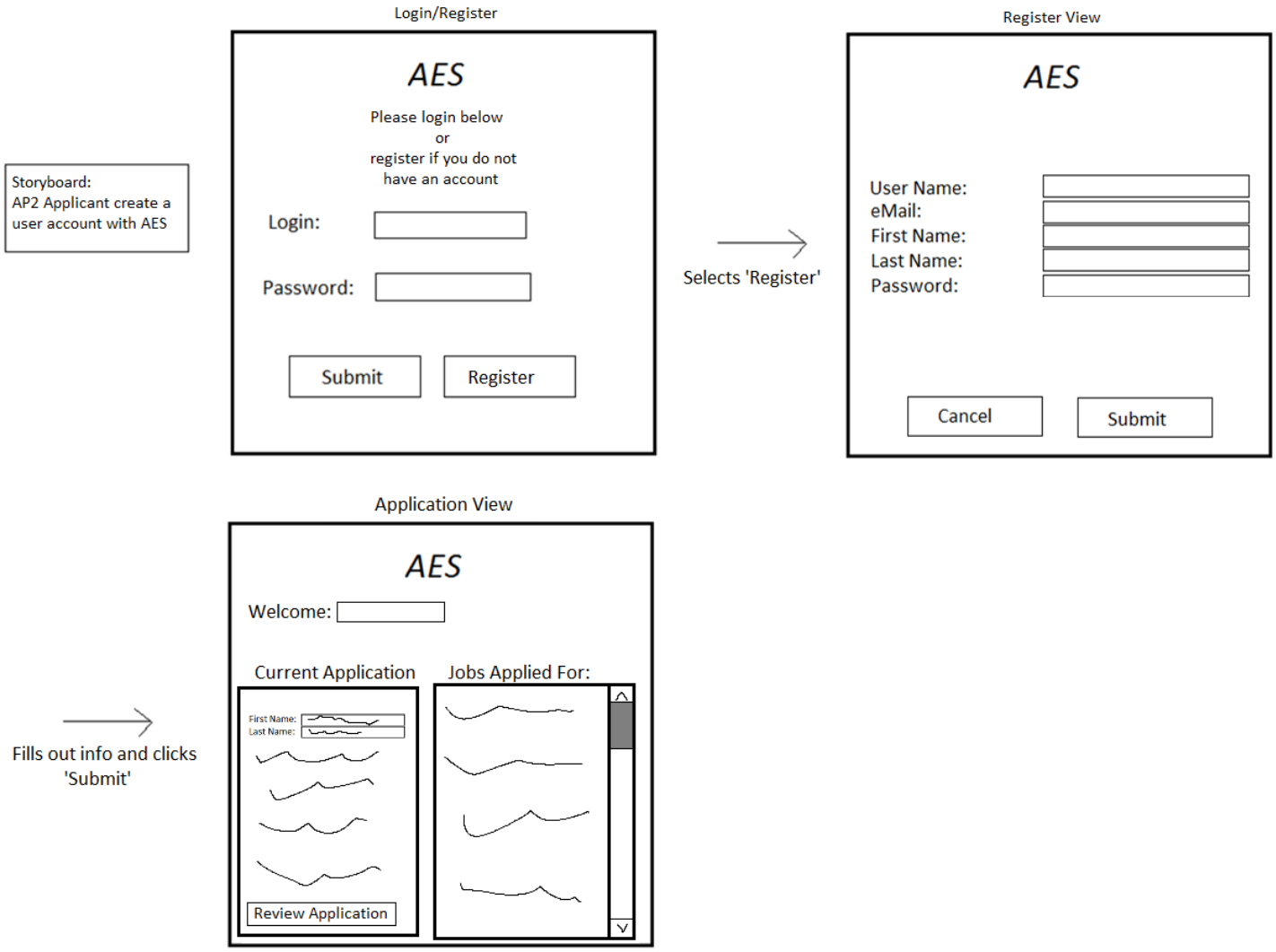


TABLE 12 - AP2 STORYBOARD

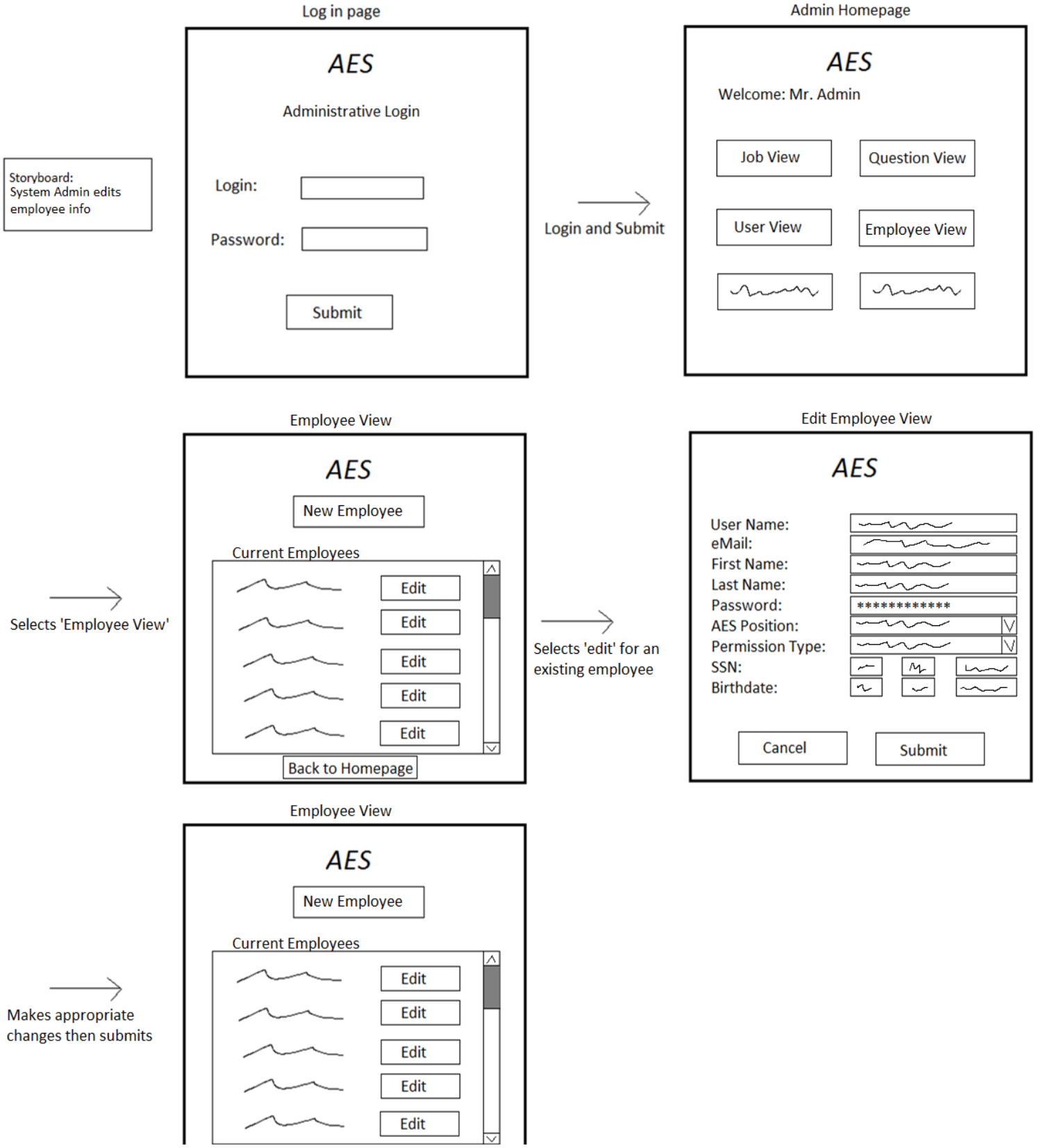


TABLE 13 - SA2 STORYBOARD

Design

A.I.M. Architecture

AIM ARCHITECTURE DOCUMENT

PURPOSE	**STRUCTURE THE SYSTEM TO SIMPLIFY DESIGN…** OVERVIEW | TECHNOLOGY | COMPONENTS | INTERATIONS
LIFECYCLE	**LIVING…** CREATED DURING PRE-PROJECT… UPDATED DURING EACH ITERATION'S DESIGN PHASE.

OVERVIEW…

The AIM system architecture is designed as a 3-tier web application …while maintaining n-tier design philosophies that allow for streamlined expansion. More specifically, AIM utilizes a web-applications tier, a web-services tier and a database tier. The web-applications tier has two components: the applicant website (AIM) and the administrative website (AIM admin). Both of these components are built on top of the ASP.NET MVC framework. Additionally, they are designed as two completely separate projects. This allows for separation of concerns (applicants vs. employees), customized user experiences and more flexible maintenance capabilities. The web-services tier is broken down into four layers, each of which are WCF based web services. The four web services (Authentication, Application, Interview and Administration) act as the intermediary between the web applications and the database. They are structured in a way that groups application/database interactions by role. For security purposes, all database interactions will be handled by these services (there are no direct website/database interactions). The database tier (AIM db) is a single SQL database hosted over a cloud infrastructure (Azure) which allows for remote access and maintenance.

COMPONENTS…

WEB-APPLICATIONS	WEB-SERVICES	DATABASE
- AIM …applicant website - AIM-admin website …administrative	- Authentication …log-in / log-out - Application screening …applicant - Interview interviewing …applicant - Administration management …back-end	- AIM-db …single database

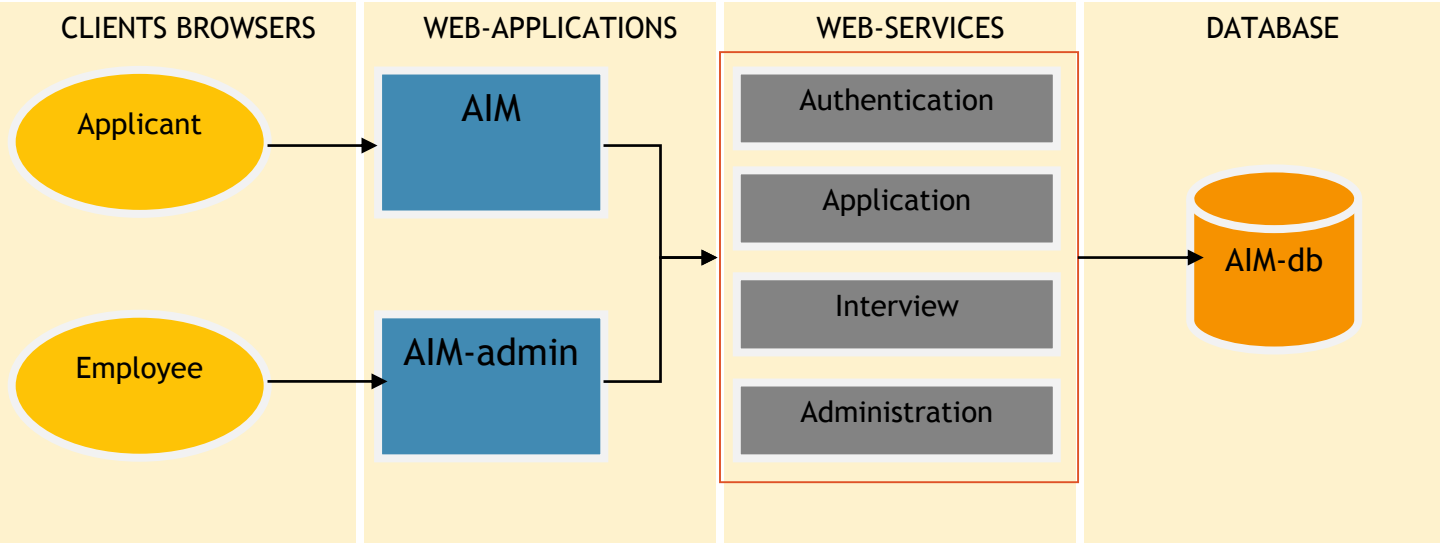
AIM:	ASP.NET (MVC) web application used by the applicants on the in-store kiosks. Will be used for all applicant interactions (such as questionnaires, applications, creating applicant accounts, logging into to existing applicant accounts and establishing phone interviews).

AIM-admin:	ASP.NET (MVC) web application used by employees and administrative personnel. Will be used for all employee interactions (such as creating and editing user accounts, creating job categories/ listings, approving job listings, conducting interviews etc).
Authentication:	WCF web-service that handles all database connections and validations involving logging in and logging out (for both the applicants and the employees.
Application:	WCF web-service that handles all database connections and validations involving the application questionnaires. For example, when the applicant applies for a job, this service retrieves the appropriate questionnaire AND handles the processing of said questionnaire to determine if the applicant can move forward in the application process. It will also be used for processing the data an applicant enters when filling out the application forms.
Interview:	WCF web-service that handles all database connections and validations involving the interview process. This will only be used by the employee web-application. Example uses include filling out interview forms while the hiring specialist conducts phone interviews. This service would then be used to save and retrieve that information.
Administration:	WCF web-service that handles all database connections and validations involving the IT administration and back-end related tasks. Examples include creating new employee user accounts, setting user permissions, creating job listings/questionnaires etc.
AIM-db:	SQL database hosted in the Azure cloud. The aforementioned services handle retrieving and saving data to and from the database.

COMPONENT TECHNOLOGY BREAKDOWN…

WEB-APPLICATIONS	WEB-SERVICES	DATABASE
FRAMEWORK: ASP.NET MVC LANGUAGES: C#, HTML / JavaScript	FRAMEWORK: WCF LANGUAGES: C#, HTML / JavaScript	FRAMEWORK: Azure /Microsoft SQL Server LANGUAGES: SQL

COMPONENT DIAGRAM…



CONCEPTUAL COMPONENT INTERFACES…		
AIM	- Display homepage - Display log-in page - Display apply-for-job page - Display create-new-user page	- Display application view - Display application page - Display questionnaire page - Display phone interview page
AIM-admin	- Display homepage (for all actors) - Display log-in page - Display new job page - Display new questionnaire page	- Display new survey question page - Display new user page - Display applicant list page - Display single applicant info page
Authentication	- Get log-in info	- Verify log-in info
Application	- Get questionnaire - Save questionnaire	- Get application - Save application
Interview	- Get applicant list - Get job /applicant list	- Get single applicant info - Save (edit) single applicant info
Administration	- Create new user - Update (edit) user info - Create job title - Update job title	- Get current jobs - Update current jobs - Get job details - Update job details
AIM-db	- Query users - Query jobs - Query survey questions - Query questionnaires	- Query job details - Query applications - Query applicants - Query applicant details

TABLE 14 - A.I.M. ARCHITECTURE DOCUMENT

TABLE 15 - INTERACTION DIAGRAM: ADMIN CREATES A NEW USER

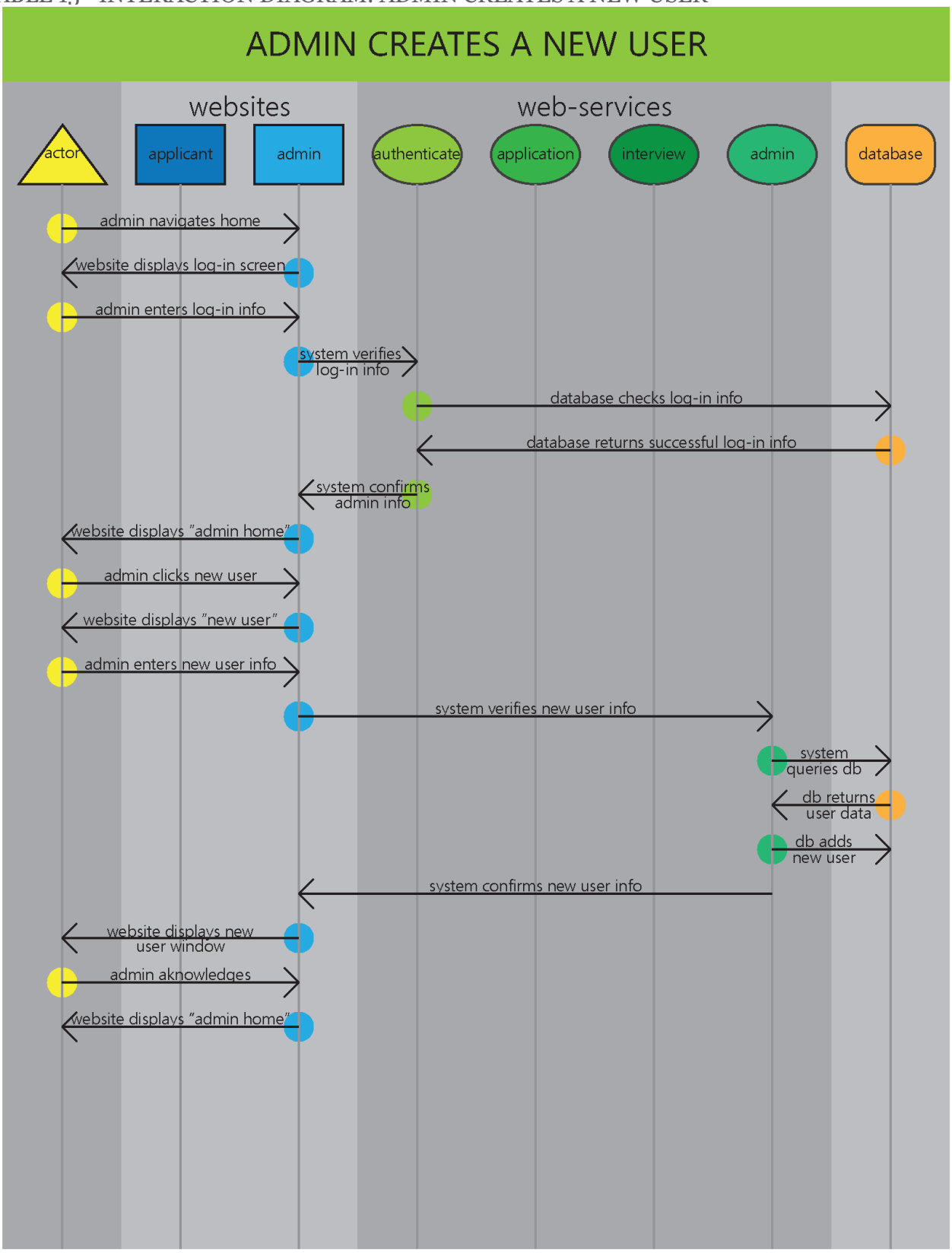
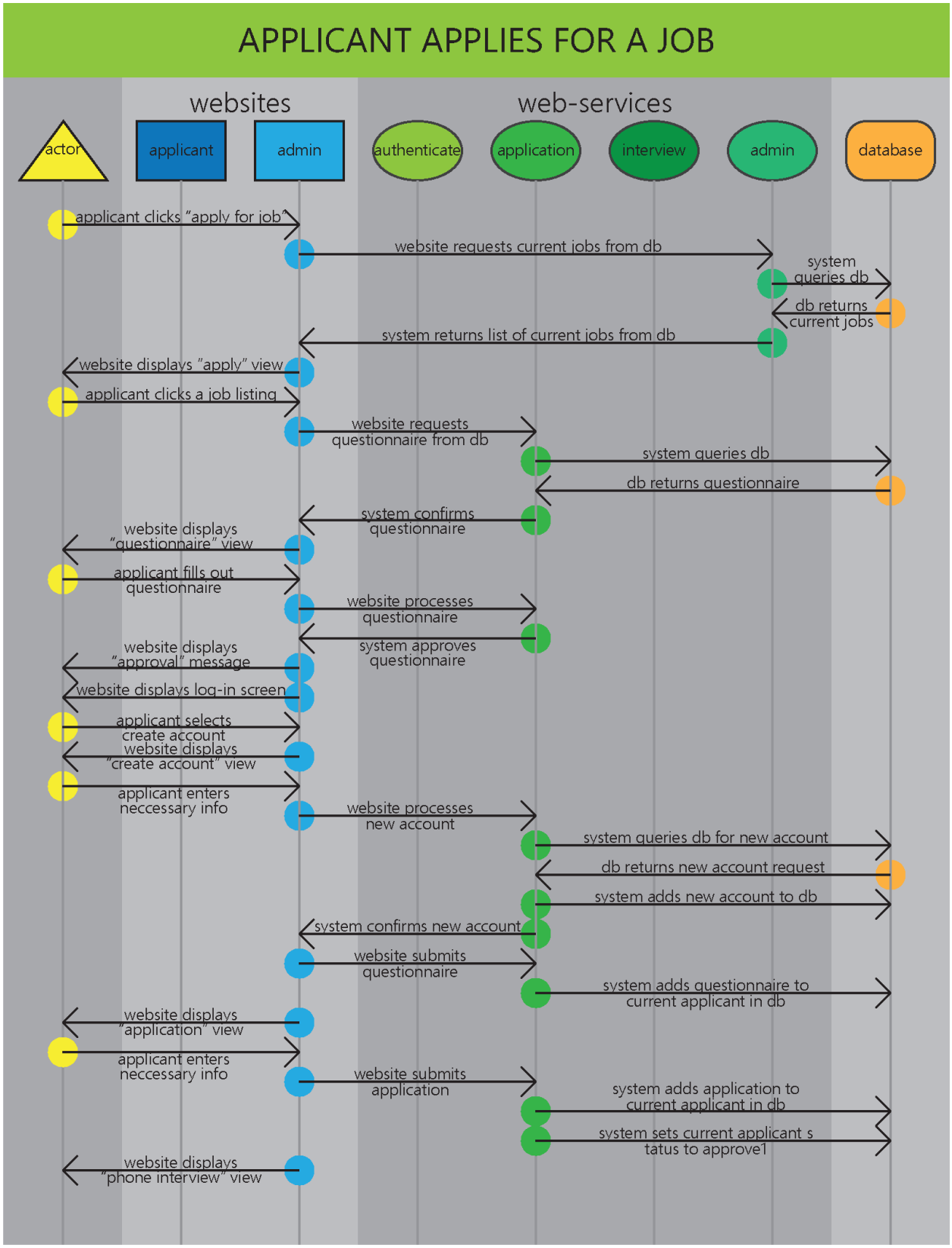


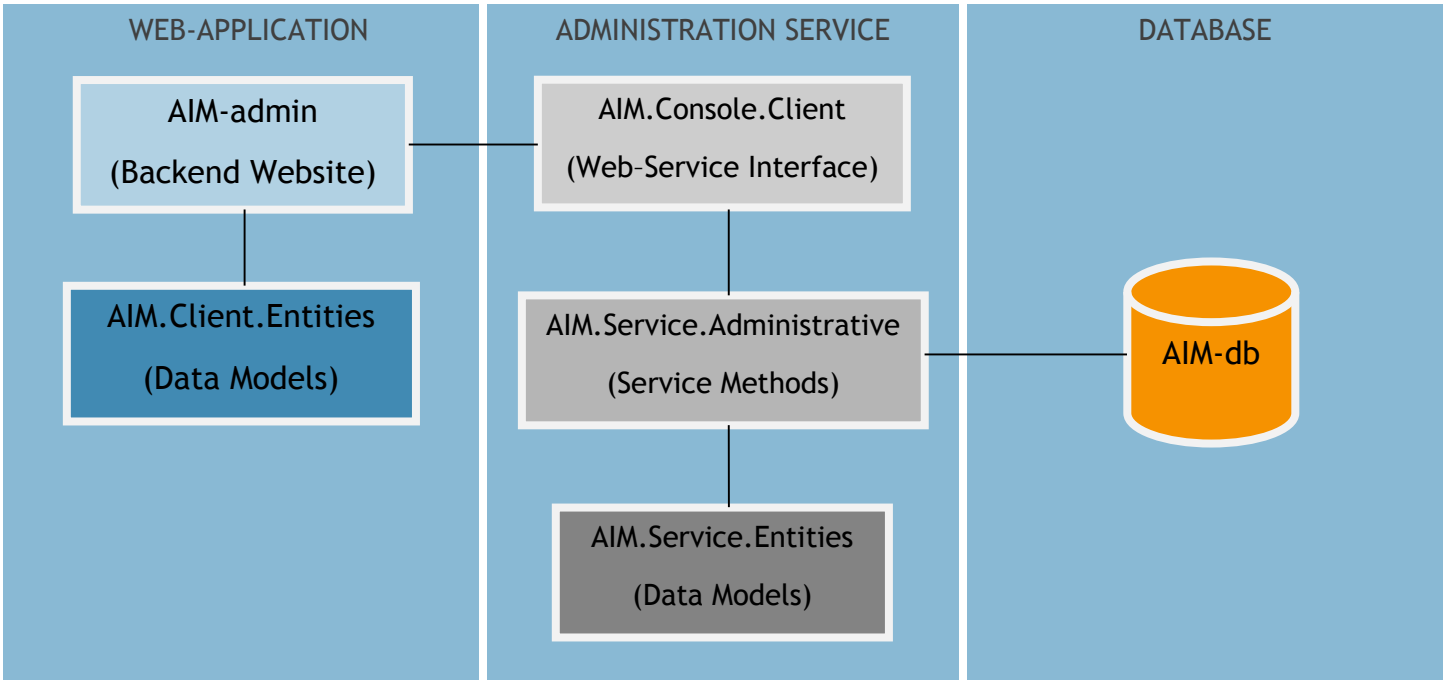
TABLE 16 - INTERACTION DIAGRAM: APPLICANT APPLIES FOR A JOB



WEB SERVICE IMPLEMENTATION…

The interaction between the client websites and a typical web service is broken down into 4 main components. The websites interact with "dummy" models that resemble the actual data within the database (These models are used only by the web applications). They can then request or push updates through an interface into the web service. The service will then include implementations of each of the methods from the interface. These methods will then modify their own versions of the database models before pushing or pulling the changes to the actual database itself (as with the website models, the service models are strictly used by the services). The purpose of storing multiple data entities is to separate client data manipulations from the physical database to create a layer of security. The interface ensures that all the necessary interactions are present and that the client has to follow the rules set by the database relationships.

SAMPLE WEB SERVICE BREAKDOWN (ADMINISTRATION SERVICE)…

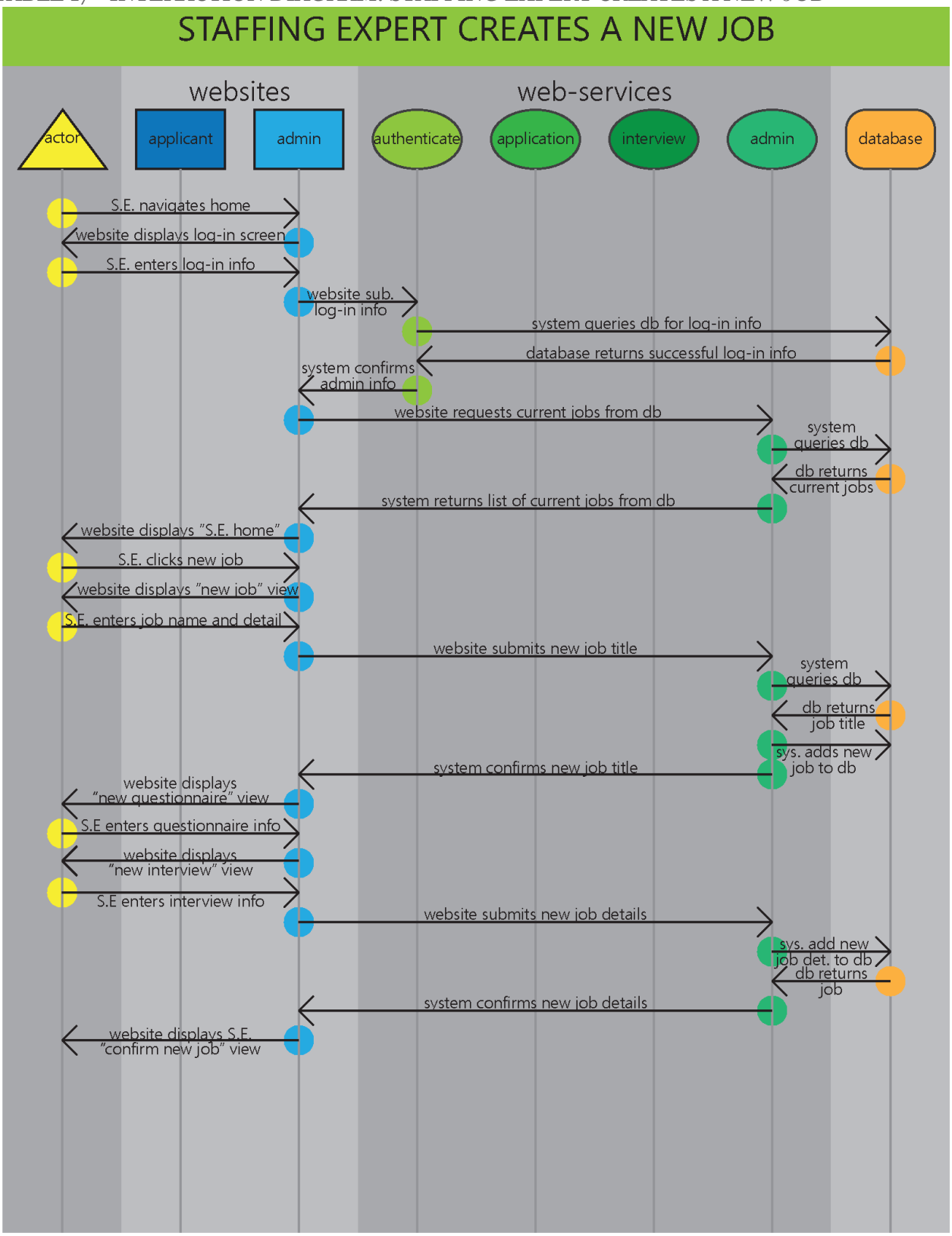


SAMPLE ADMINISTRATIVE SERVICE INTERFACE METHODS…

IJobService	Task<IEnumerable<Job>> GetJobsList(); Task<Job> GetJob(int id); Task<Job> UpdateJob(Job entity); Task<Job> CreateJob(Job entity); Task<bool> DeleteJob(int id);
IPersonalInfoService	Task<IEnumerable<PersonalInfo>> GetPersonalInfoes(); Task<PersonalInfo> GetPersonalInfo(int id); Task<PersonalInfo> UpdatePersonalInfo(PersonalInfo entity);

	Task<PersonalInfo> CreatePersonalInfo(PersonalInfo entity); Task<bool> DeletePersonalInfo(int id);
IQuestionService	Task<IEnumerable<Question>> GetQuestionsList(); Task<Question> GetQuestion(int id); Task<Question> UpdateQuestion(Question entity); Task<Question> CreateQuestion(Question entity); Task<bool> DeleteQuestion(int id); Task<IEnumerable<Questionnaire>> GetQuestionnairesList(); Task<Questionnaire> GetQuestionnaire(int id); Task<Questionnaire> UpdateQuestionnaire(Questionnaire entity); Task<Questionnaire> CreateQuestionnaire(Questionnaire entity); Task<bool> DeleteQuestionnaire(int id);
IUserService	Task<IEnumerable<User>> GetUsersList(); Task<User> GetUser(int id); Task<User> UpdateUser(User entity); Task<User> CreateUser(User entity); Task<bool> DeleteUser(int id);

TABLE 17 - INTERACTION DIAGRAM: STAFFING EXPERT CREATES A NEW JOB



A.I.M. Design Document

Name: "A.I.M. Design Iteration 1"
Purpose: The AIM system design is as a 3-tier web application …while maintaining n-tier design philosophies that allow for streamlined expansion. More specifically, AIM utilizes a web-applications tier, a web-services tier and a database tier. The web-applications tier has two components: the applicant website (AIM) and the administrative website (AIM admin). Both of these components are built on top of the ASP.NET MVC framework. Additionally, they are designed as two completely separate projects. This allows for separation of concerns (applicants vs. employees), customized user experiences and more flexible maintenance capabilities. The web-services tier is broken down into four layers, each of which are WCF based web services. The four web services (Authentication, Application, Interview and Administration) act as the intermediary between the web applications and the database. They are structured in a way that groups application/database interactions by role. For security purposes, all database interactions will be handled by these services (there are no direct website/database interactions). The database tier (AIM db) is a single SQL database hosted over a cloud infrastructure (Azure) which allows for remote access and maintenance.
Living: Created during first iteration. Updated during each iteration's design phase.
Sections:

A.I.M. Iteration 1 Use Cases

ID#	Use Case #	Description	Need	Risk	Time Est. (In Days)	Week planned	Additional Notes	Goals
50	Architecture 1st Iteration	Design and implement the Database, create basic DB models, create ASP.NET MVC application, create WCF services, create interface models, and create relationships and connect all services	Done	Done	Done	Done		Architecture frame completed
51	Architecture / Admin UI	Designing and implementing the administrative UI that all actors will use portions of	Done	Done	Done	Done		Administrative UI and functionality complete
16	SA1	System Admin adds new employee to system	Done	Done	Done			User login system created
17	SA1 Alt1	System Admin removes employee from system	Done	Done	Done	Done		Staffing Expert functionality to create/edit/delete job list and attached questionnaire system complete
18	SA2	System Admin edits employee information	High	Low	1.5			
19	SA3	System Admin changes permission level for employee	Med	Low	1.5	Week 9		
20	SA4	System Admin fixes typo in questionnaire	Low	Low	2			
35	SE1	Staffing expert creates job title and description	Done	Done	Done	Done		

36	SE2	Staffing expert creates questions to be used in questionnaires	**High**	**Low**	1.5			
37	SE3	Staffing expert creates questionnaire for a job	**High**	**Med**	3	Week 10		
38	SE4	Staffing expert edits a question from pool	**Low**	**Low**	2			
39	SE5	Staffing expert edits a job	Done	Done	Done			
45	NF1	Will be used by English speaking adults	Done	Done	Done			
49	NF5	Assumed hardware used is Computer, Keyboard, and Mouse	Done	Done	Done			
48	NF4	Disability accommodation is currently not needed	Done	Done	Done	Done		
		End of Iteration 1			**11.5**			

A.I.M. Iteration 1 Use Case Diagrams

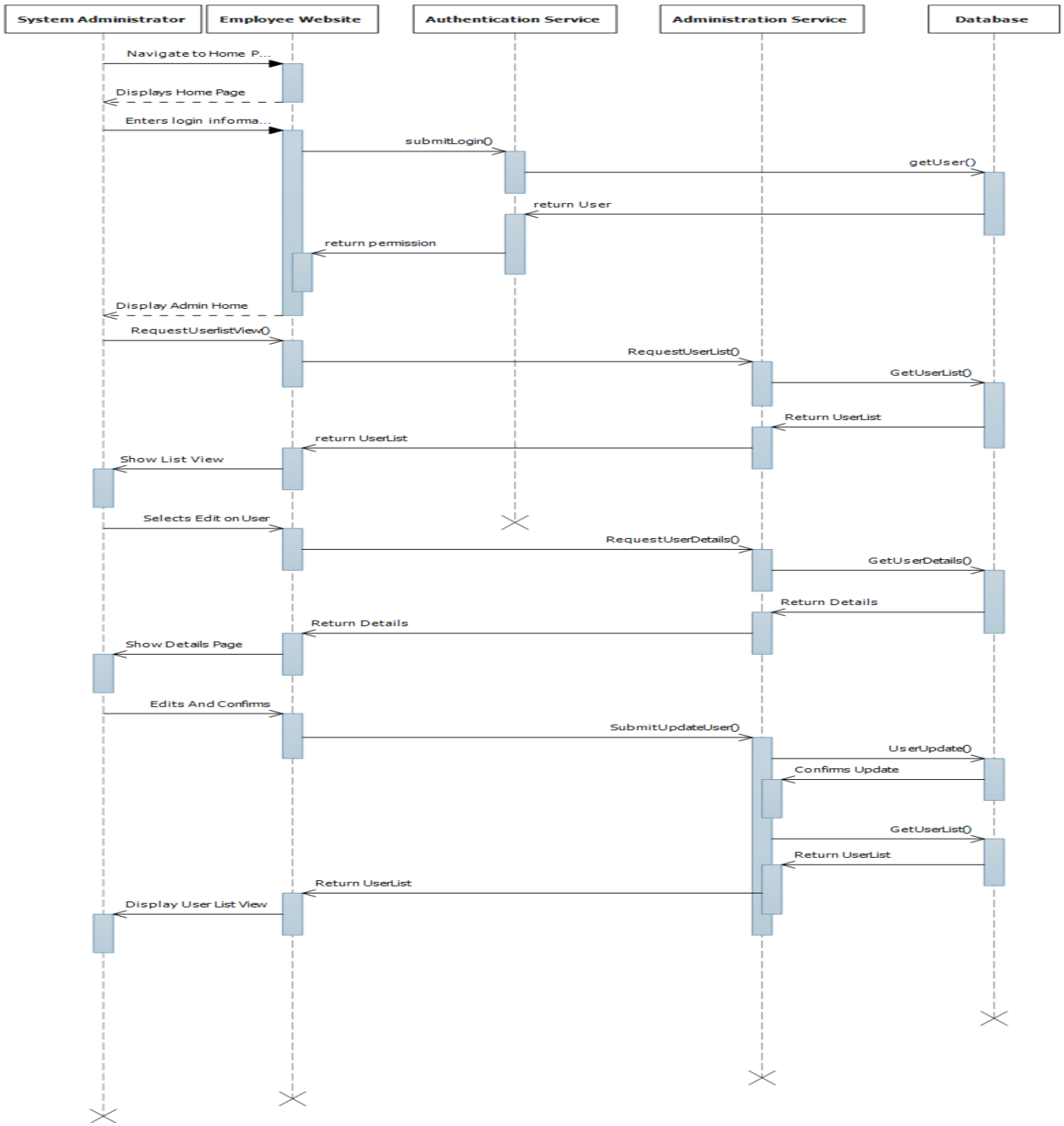


TABLE 18 - SA2 SEQUENCE DIAGRAM

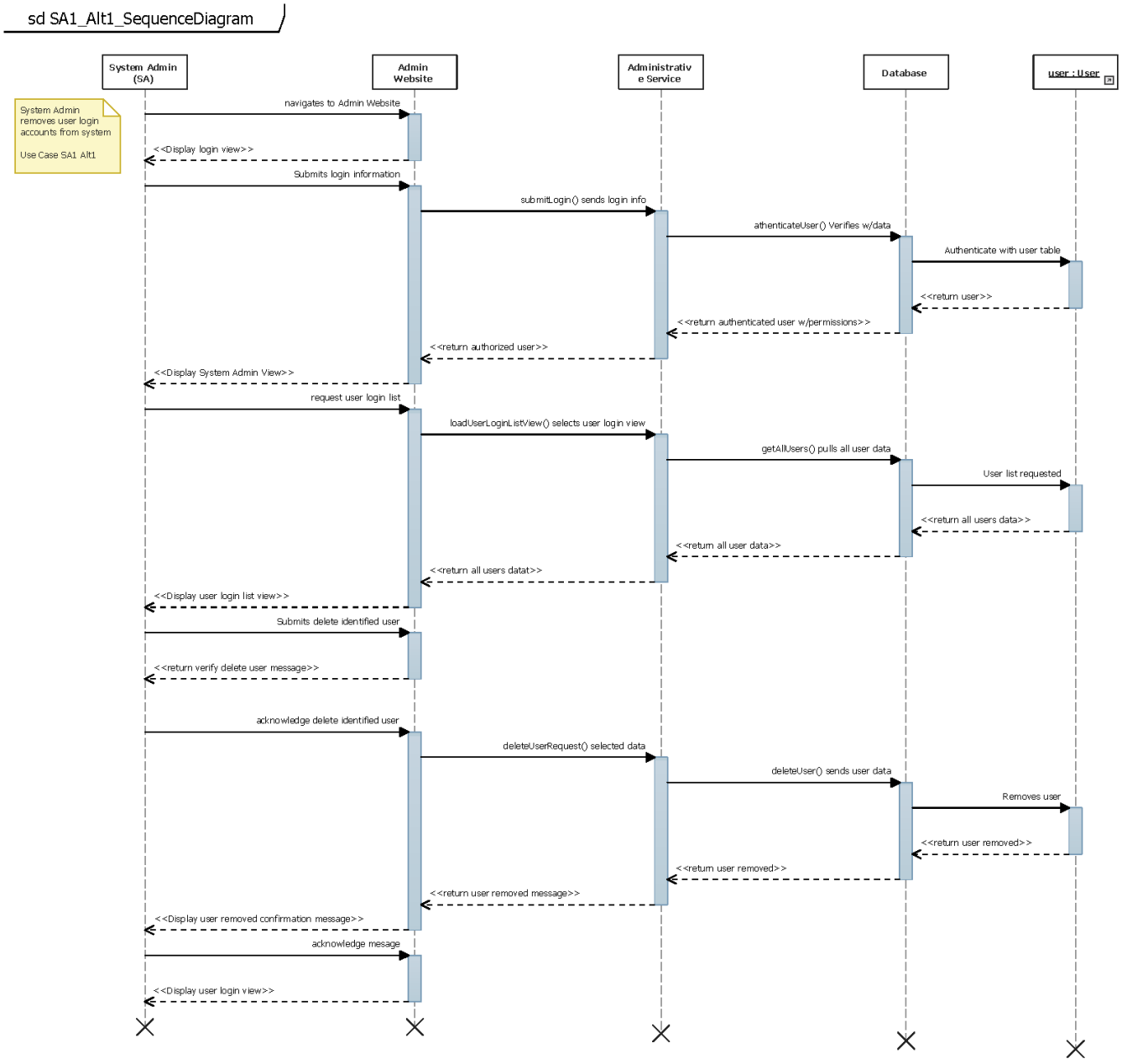


TABLE 19 - USE CASE SA1 ALT1

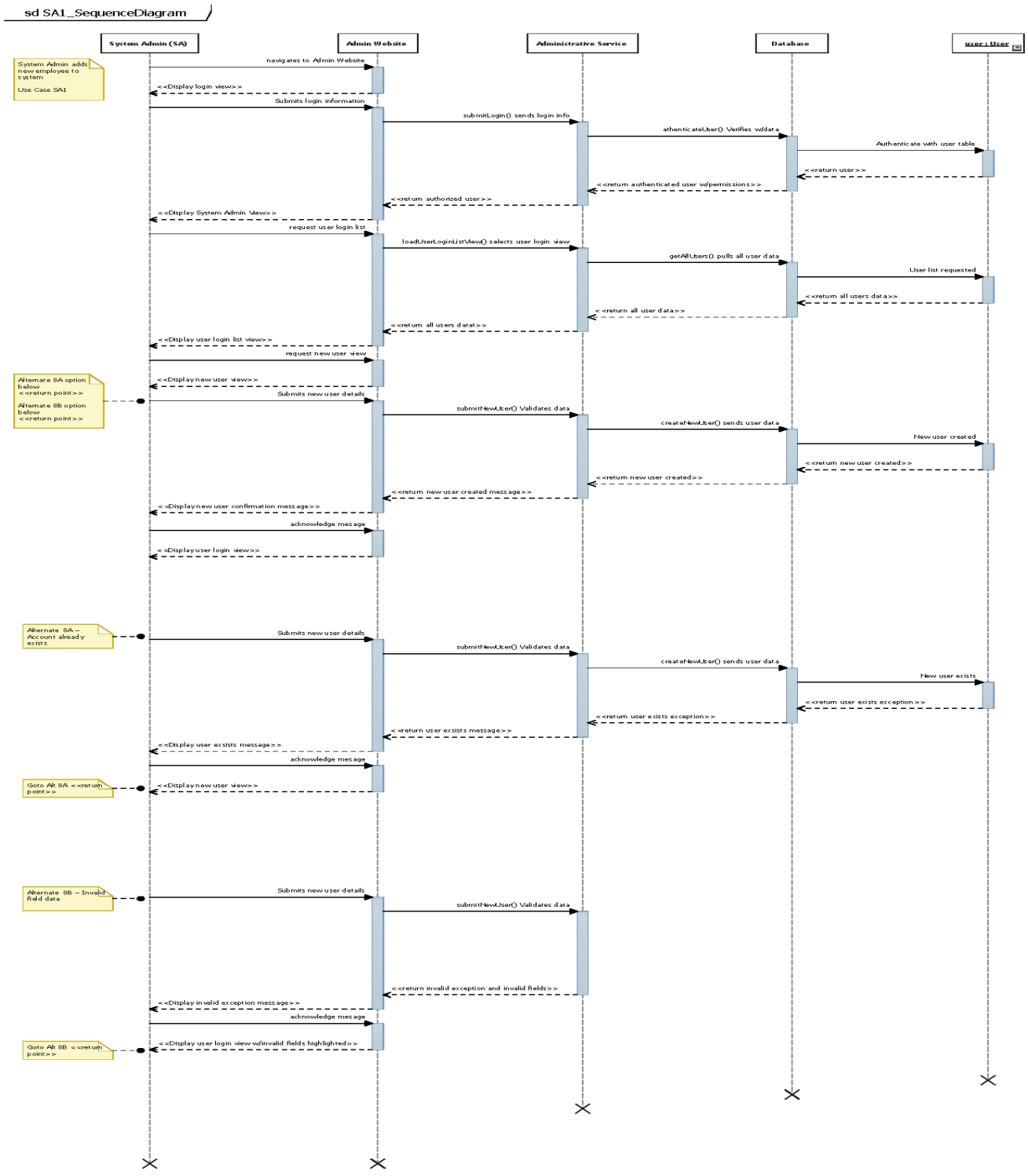


TABLE 20 - USE CASE SA1

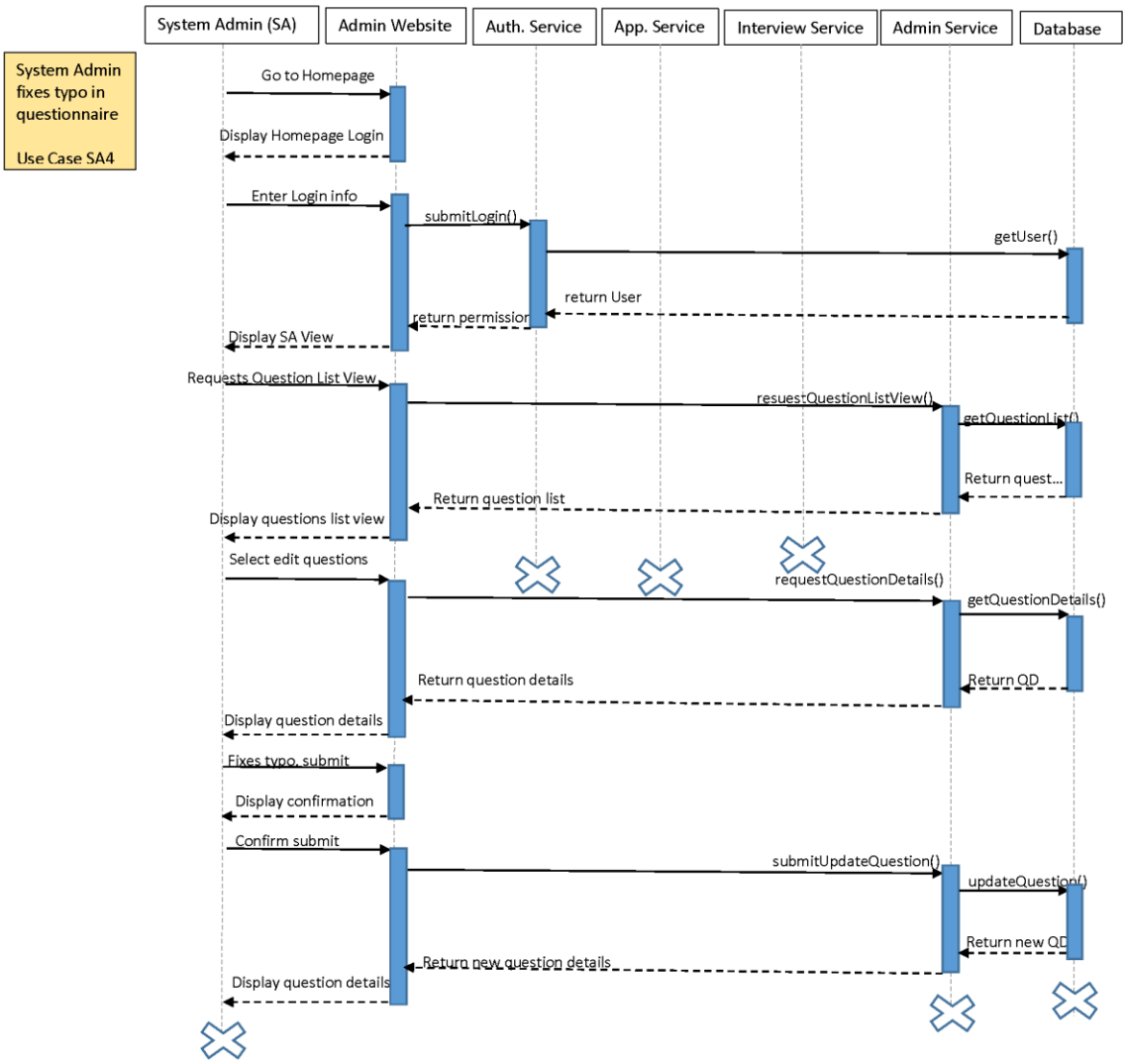


TABLE 21 - USE CASE SA4

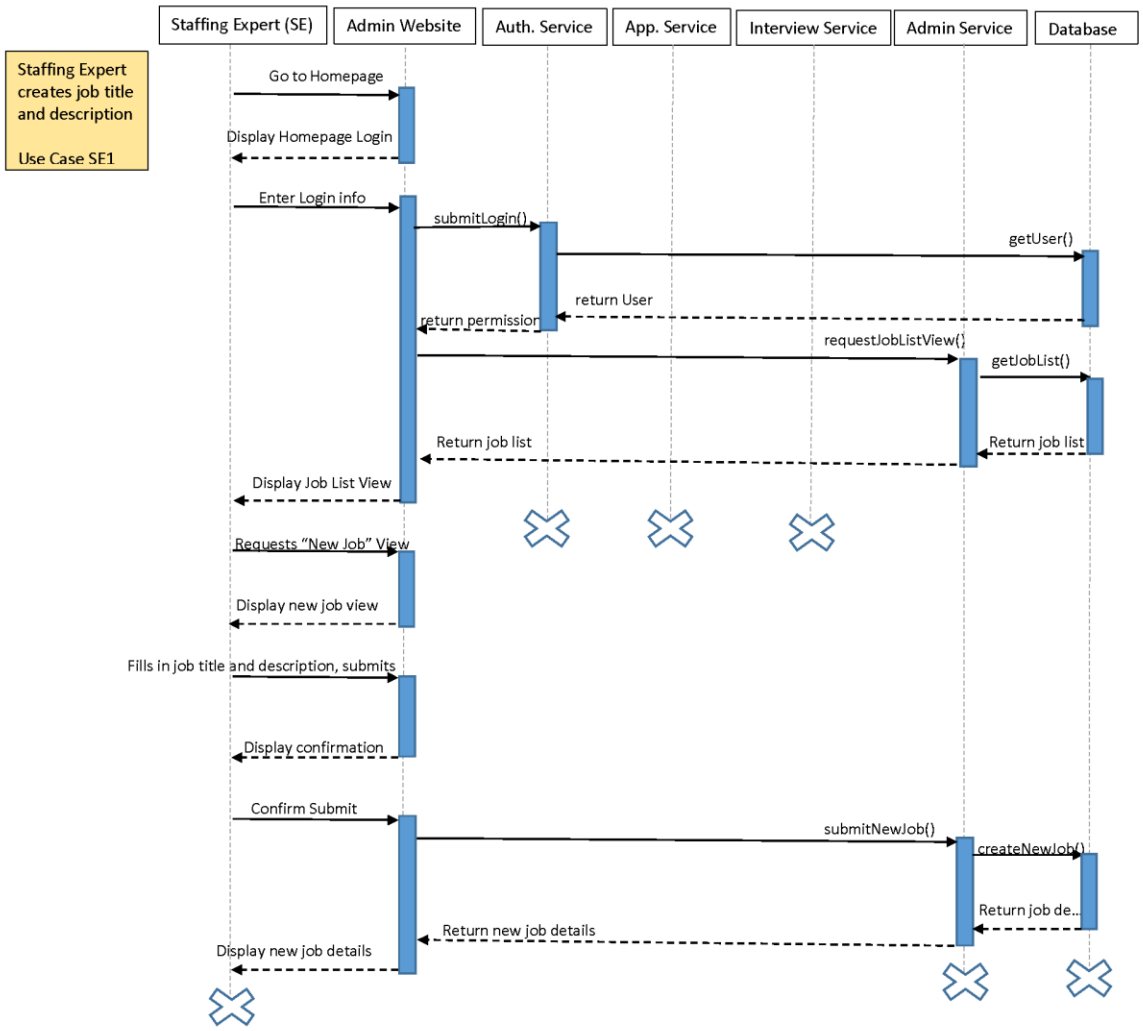


TABLE 22 - USE CASE SE1

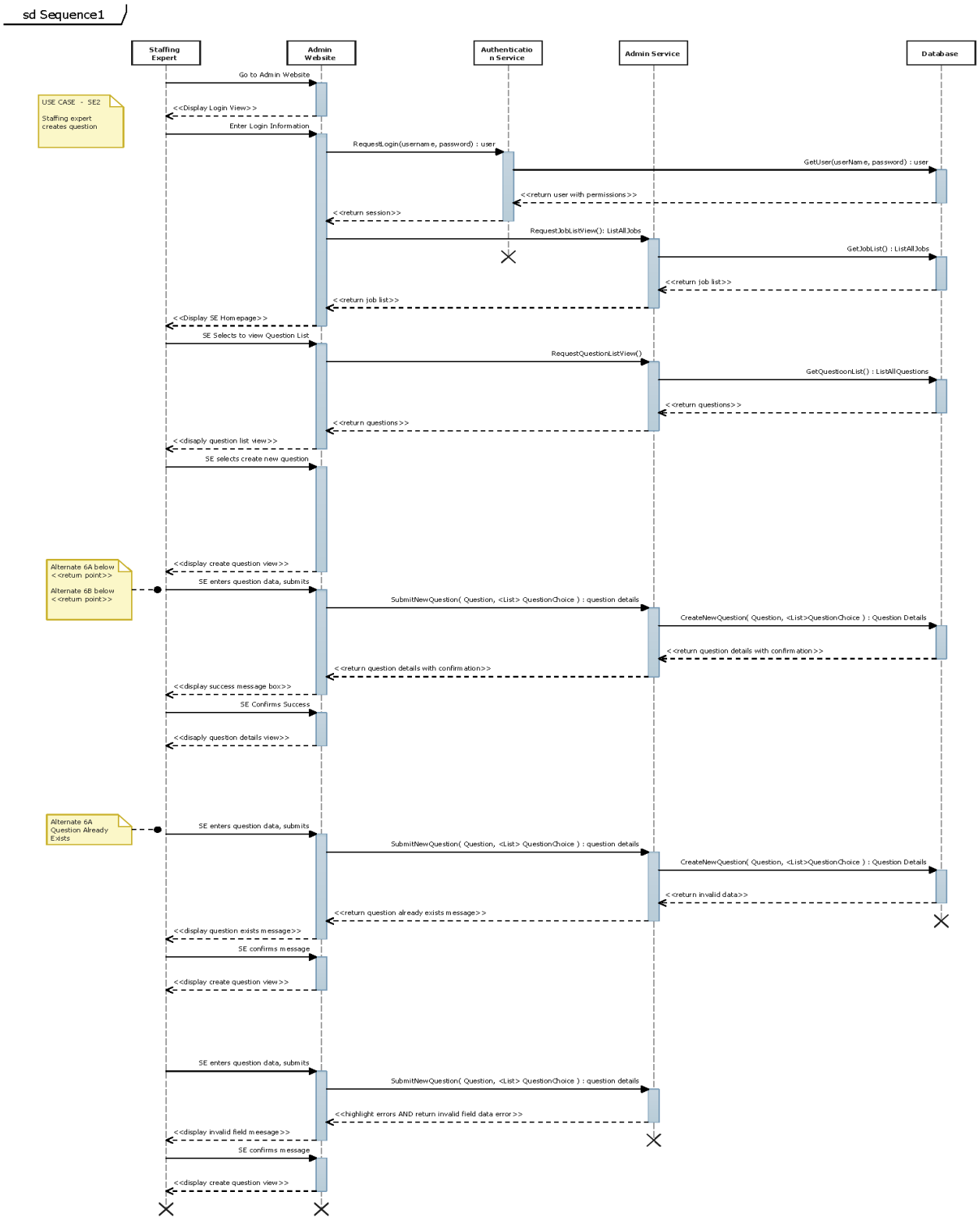


TABLE 23 - USE CASE SE2 UPDATED

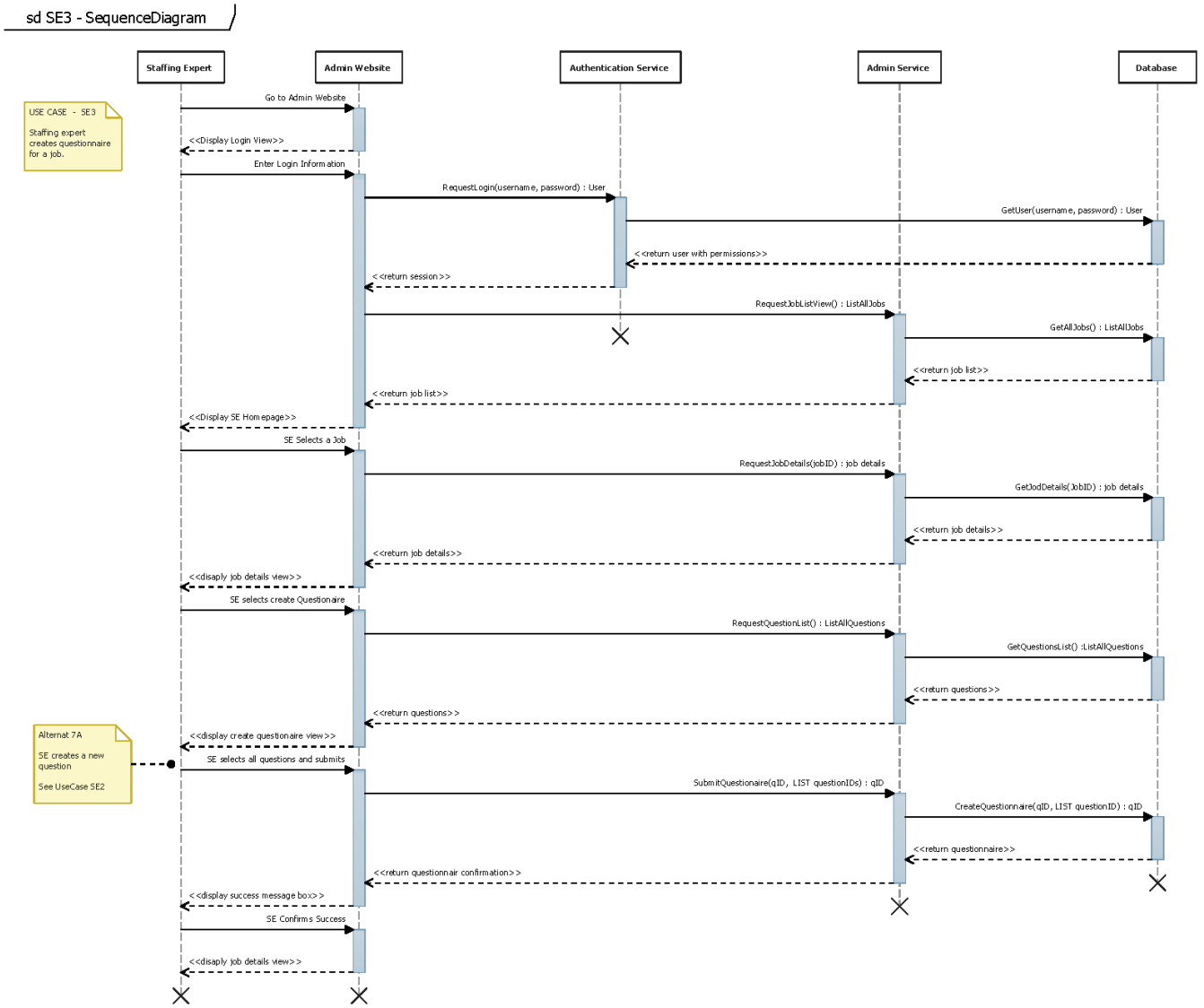


TABLE 24 - USE CASE SE3 UPDATED

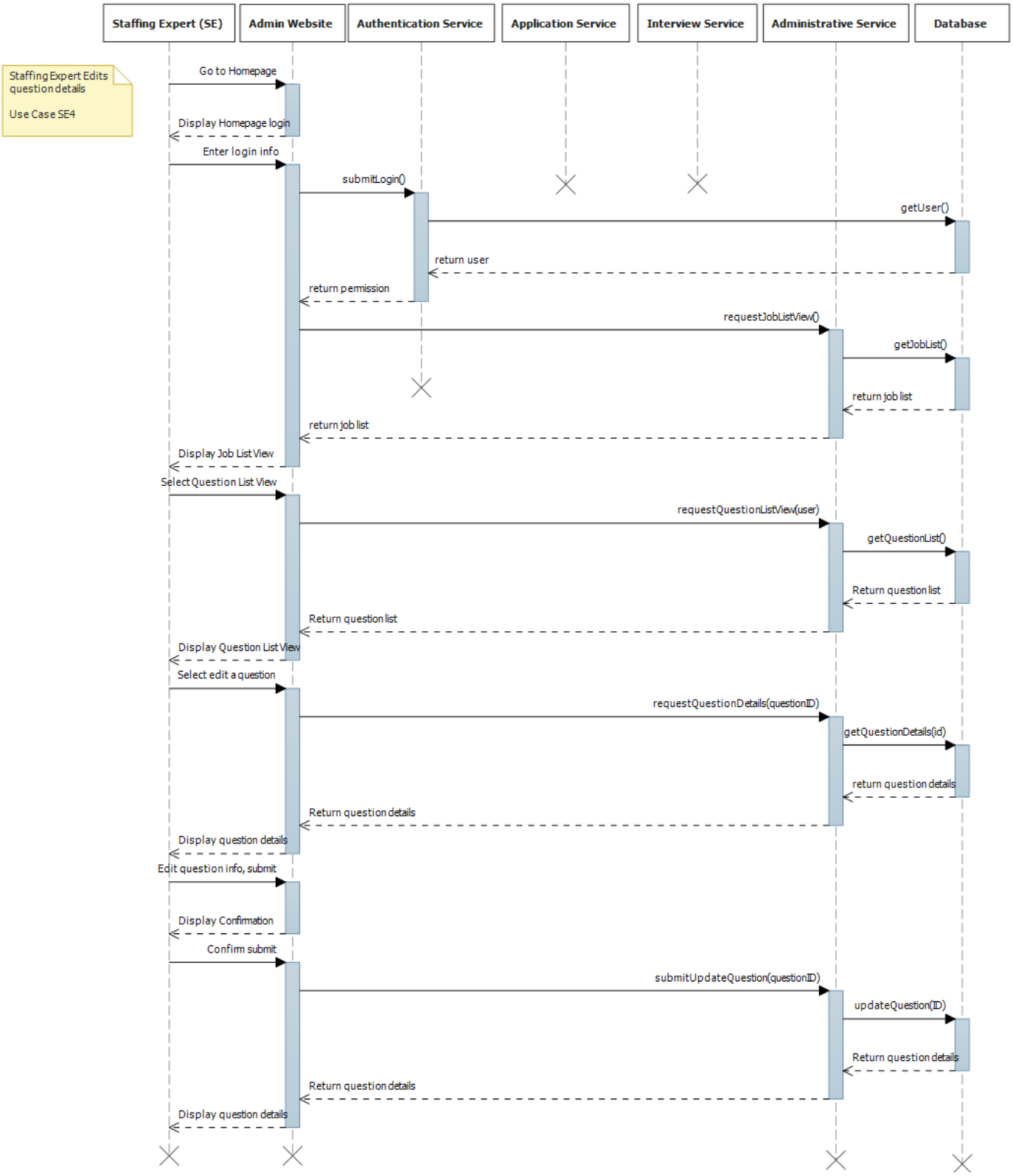


TABLE 25 - USE CASE SE4

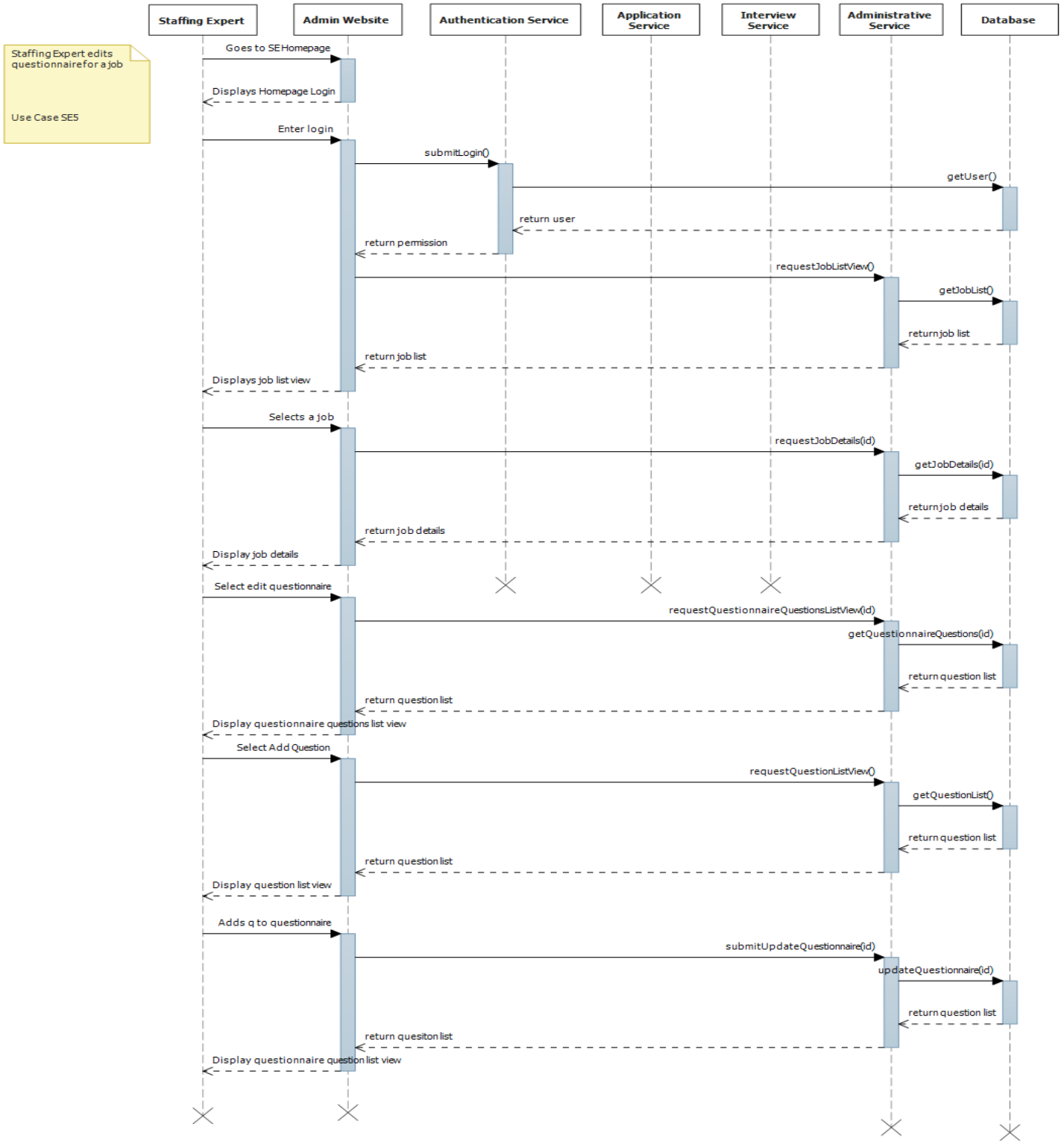


TABLE 26 - USE CASE SE5

A.I.M. Database Model

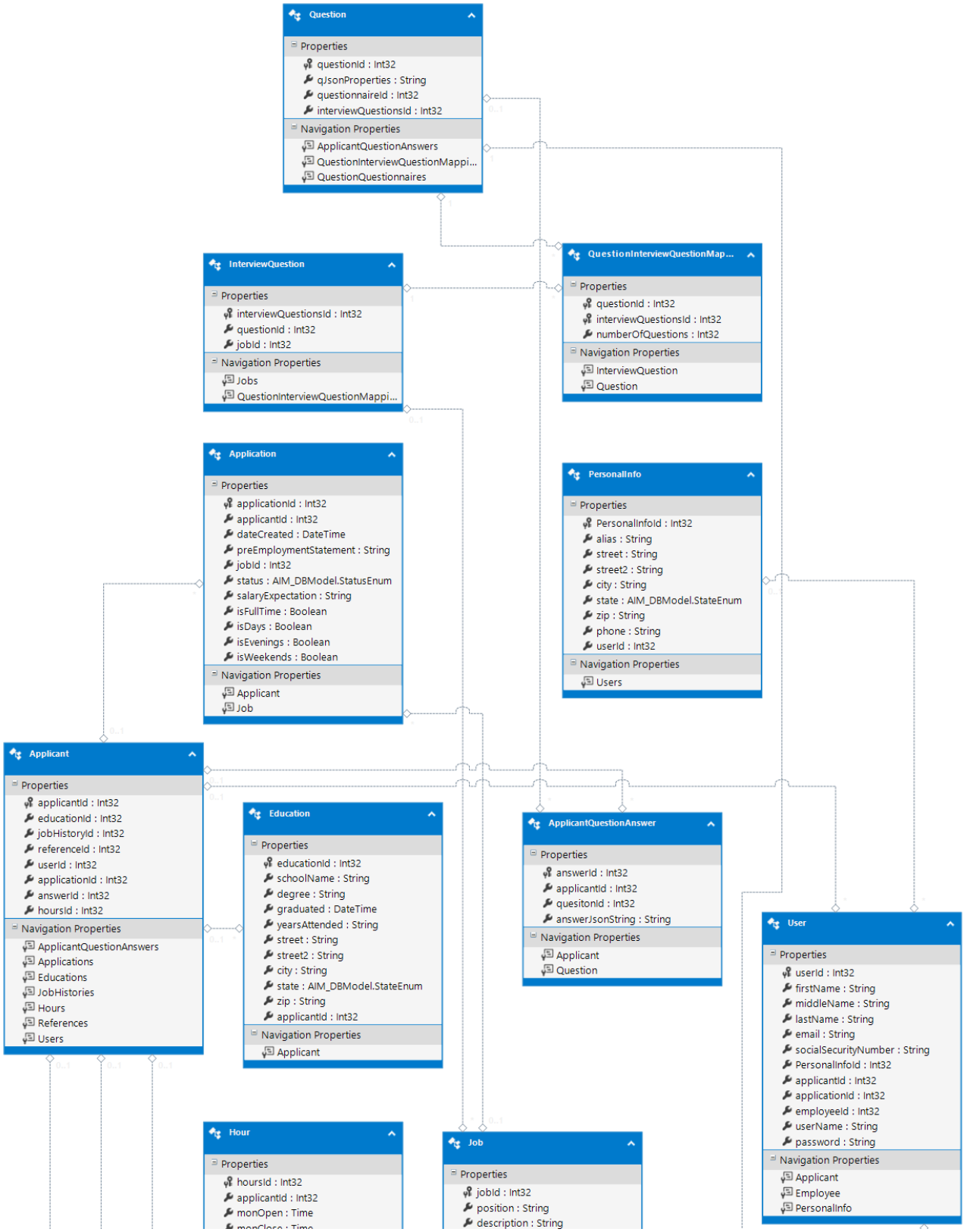
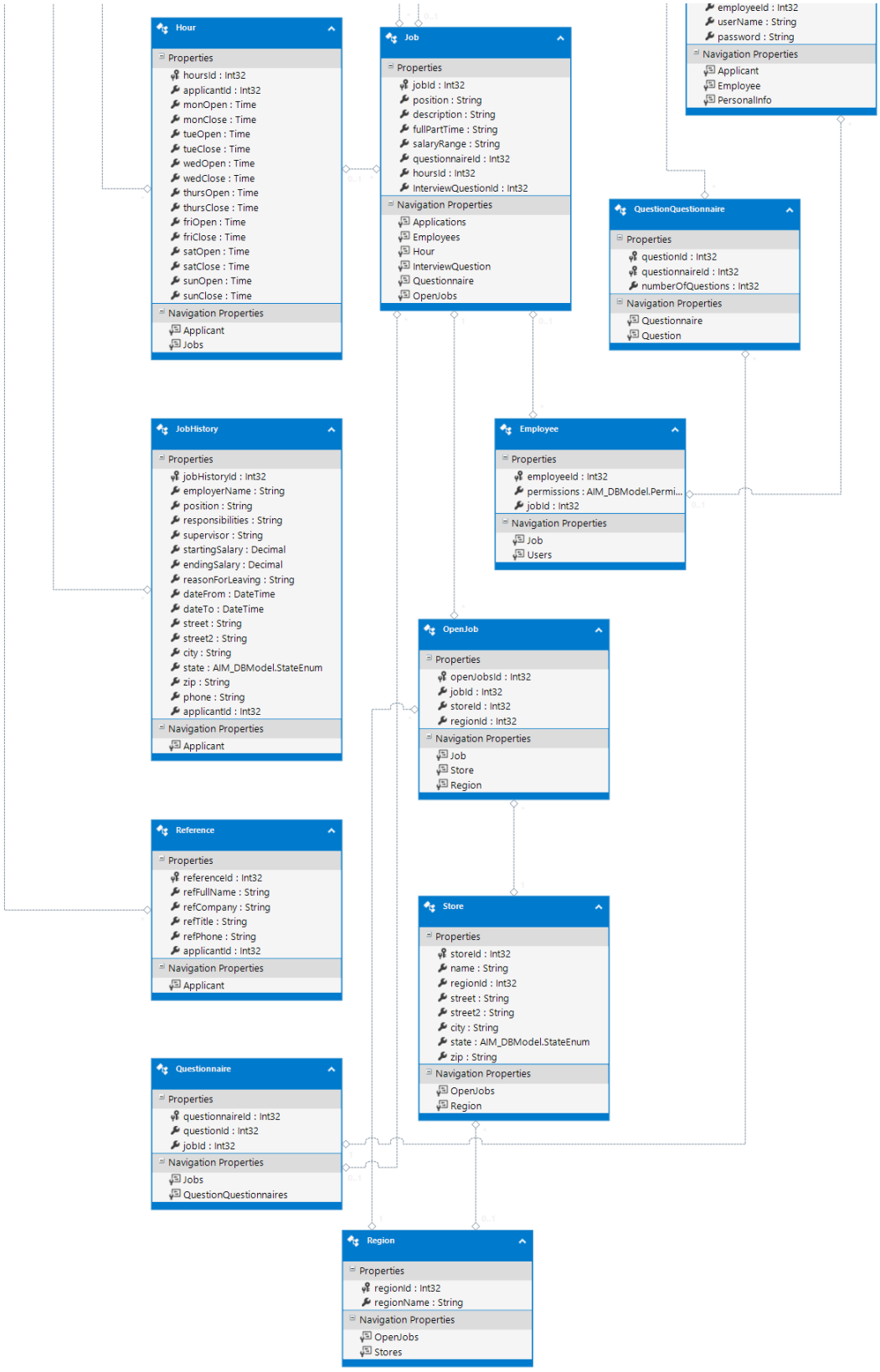


TABLE 27 DATABASE MODEL



A.I.M. Controllers and Method Stubs

JOBCONTROLLER:
classes involved: Job, DBContext, Questionnaire

```
public class JobController : Controller
{
        private DBContext db = new DBConstext();


        Index():view
        // GET: /Job/
        public ActionResult Index(string sortOrder)
        {
                return View(job.ToList());
        }
        DetailJobView():view
        // GET: /Job/Details/5
        public ActionResult Details(int? id)
        {
                return View( job );
        }
        CreateJobView():view
        // GET: /Job/Create
        public ActionResult Create()
        {
                return View(job);
        }
        CreateJobView():view
        // POST: /Job/Create
        public ActionResult Create([Bind(Include="JobId, … , …, …, …, …")] Job job)
        {
                return View(job);
        }
```

- requestJobListView():all jobs

- submitNewJob(jobID): job details

- requestJobDetails(jobID): job details

- createQuestionnaireView(): all questions

- editQuestionnaireView(questionnaireID): Questionnaire.Include (Question)

- submitNewQuestionnaire(questionnaire): Questionnaire.Include(Question)

QUESTIONCONTROLLER:
classes involved: Question, DBContext

```
public class QuestionController : Controller
{
        private DBContext db = new DBContext();
        Index():view
        // GET: /Question
        public ActionResult Index( )
        {
                return View( question.ToList() );
        }

        QuestionListView():view
        // GET: /Question
        public ActionResult Details( )
        {
                return View(question.ToList());
        }

        DetailQuestionView():view
        // GET: /Question/Details/5
        public ActionResult Details(int? id)
        {
                return View(question);
        }
        CreateQuestionView():view
        // POST: /Question/Create
        public ActionResult Create([Bind(Include="QuestionId, … , …, …, …, …")] Question question)
        {
                return View(question);
        }

        public List<Question> requestQuestionList()
        {
                return db.getQuestionList();
        }


        public QuestionDetailsClass requestQuestionDetails(QuestionId)
        {
                return db.getQuestionDetails(question);
        }
```

```
public QuestionDetailsClass submitUpdateQuestion(Question question)
{
        db.updateQuestion(question);


        return db.getQuestionDetails(question);
}

public List<Question> submitNewQuestion(Question question)
{
        db.createNewQuestion(question);

        return db.getQuestionList();
}
```

ADMIN CONTROLLER

```
public class AdminController : Controller
{
// Classes involved: User, DBContext

        DBContext db = new DBContext(); // Whatever the syntax is...

        #region Various Views
        public ActionResult Index() { return View(); } *Done*

        public ActionResult CreateUserView() { return View(); } *Done*

        public ActionResult Create([Bind(stuff)]) *Prebuilt by VS*
        //Post
        {
                return View();
        }

        public ActionResult RequestUserListView()  { return View(); } *Done*

        public ActionResult DetailUserView( int? userID ) { return View(user); } *Done*

        public ActionResult EditDetailUserView(int? userID) { return View(user); } *Done*
        #endregion

        public List<User> submitNewUser(int? user)
        {
                db.createNewUser(user);

                return db.getUserList();
        }

        public List<User> requestDeleteUser(int? userID) *Done?*
        {
```

```
            db.deleteUser(userID);

            return db.getUserList();
        }

        public UserDetailsClass requestUserDetails(int? userID) *Done?*
        {
            return db.getUserDetails(user);
        }

        public UserDetailsClass submitUpdateUser(User user)
        {
            db.submitUpdateUser(user);


            return db.getUserDetails(user);
        }


}
```

ADMINISTRATIVE SERVICE

```
// Administrative web service methods

        GetUserList()
            {
                // return all users
                var userlist = db.User.OrderBy(i => i.name) ;
                return (   userlist.ToList()  ) ;
            }


        AddUser( [ Bind ( Include = "UserID, userName, firstName, middleName,  lastName, password,
        passwordCoder" ) ] User user)
            {
                // add new user to database
                db.User.Add( user );
                db.SaveChanges();


                // returns all users
                var userlist = db.Users.OrderBy(i => i.name) ;
                return (   userlist.ToList()  ) ;
            }


        DeleteUser(  int? userID  )
            {
                // delete user from database
                User user = db.User.Find( userID );
```

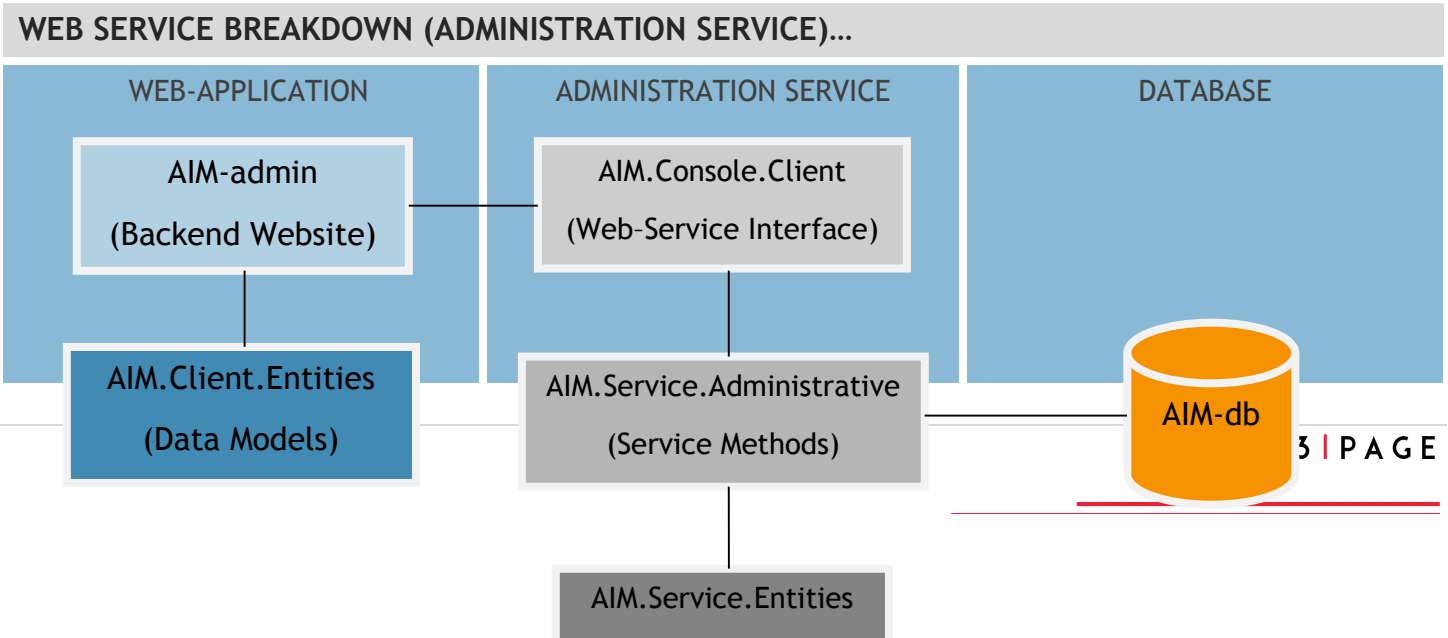
```
                db.User.Remove( user );
                db.SaveChanges();


                // returns all users
                var userlist = db.Users.OrderBy(i => i.name) ;
                return (   userlist.ToList()  ) ;
        }

GetUser(  int? userID  )
        {
                // get user details
                User user = db.User.Find( userID );
                return ( user );
        }

UpdateUser(  int? userID  )
        {
                // update existing user
                User user = db.User.Find( userID );
                db.Entry( user ).State = EntityState.Modified;
                db.SaveChanges();
                return( user );
        }

getQuestionList()
        {
                // return all questions
                var questionlist = db.Question.OrderBy( i => i.questionID ) ;
                return (   questionlist .ToList()  ) ;
        }

getQuestionDetails(  int? questionID  )
        {
                // get question details
                Question question = db.Question.Find( questionID );
                return ( question );
        }

updateQuestion(  int? questionID )
        {
                // update existing question
                Question question= db.Question .Find( questionID );
                db.Entry( question).State = EntityState.Modified;
                db.SaveChanges();
                return( question );
        }
```

```
createNewQuestion( [ Bind ( Include = ”questionID, questionText, questionType,
questionOptionString, questionAnswerString” ) ] Question question ) )
        {
                // add new question to database
                db.Question .Add( question );
                db.SaveChanges();
                return( question );
        }

getJobList( jobID ):all jobs

createNewJob(  job ):  job details

getJobDetails(jobID): job details

createNewQuestionnaire( questionnaire [ include all var ] ): questionnaire

getQuestionnaire( id ): questionnaire.Include(questions)
```

LOGINCONTROLLER:
classes involved: user, DBContext

```
public class LoginController : Controller
{
        private DBContext db = new DBContext();

        submitLogin(username, password): redirectView
        {
                return view(); //the redirect view
        }
}
```

A.I.M. Services Design



SAMPLE ADMINISTRATIVE SERVICE INTERFACE METHODS…

IJobService	Task<IEnumerable<Job>> GetJobsList(); Task<Job> GetJob(int id); Task<Job> UpdateJob(Job entity); Task<Job> CreateJob(Job entity); Task<bool> DeleteJob(int id);
IPersonalInfoService	Task<IEnumerable<PersonalInfo>> GetPersonalInfoes(); Task<PersonalInfo> GetPersonalInfo(int id); Task<PersonalInfo> UpdatePersonalInfo(PersonalInfo entity); Task<PersonalInfo> CreatePersonalInfo(PersonalInfo entity); Task<bool> DeletePersonalInfo(int id);
IQuestionService	Task<IEnumerable<Question>> GetQuestionsList(); Task<Question> GetQuestion(int id); Task<Question> UpdateQuestion(Question entity); Task<Question> CreateQuestion(Question entity); Task<bool> DeleteQuestion(int id); Task<IEnumerable<Questionnaire>> GetQuestionnairesList(); Task<Questionnaire> GetQuestionnaire(int id); Task<Questionnaire> UpdateQuestionnaire(Questionnaire entity); Task<Questionnaire> CreateQuestionnaire(Questionnaire entity); Task<bool> DeleteQuestionnaire(int id);
IUserService	Task<IEnumerable<User>> GetUsersList(); Task<User> GetUser(int id); Task<User> UpdateUser(User entity); Task<User> CreateUser(User entity); Task<bool> DeleteUser(int id);

Quality Assurance

A.I.M. Quality Assurance Plan

Name: A.I.M. Quality Assurance
Purpose: To outline and define the methods used to ensure a well-made and functioning system.
Living: Created during the pre-project, updated during each iteration.

Overview:

The goal is to ensure the highest quality product possible by utilizing testing throughout development. This will be done by creating a separate project that will be used to test all of the functionality of the program. Unit testing, Integration testing, and (maybe) System testing will all be included in this project. Github will be utilized to help prevent the merging of defective code, and will also allow us to all work both separately and together. Where appropriate, test driven development will be utilized. These tests will help to reinforce the requirements.

Goals:

- Ensure thorough and meaningful testing by making the test cases based on requirements.
- Ensure a high quality product by thorough testing and using well-built tests.
- Utilize Visual Studios testing resources to use while coding.
- Utilize test driven development when appropriate for requirements.
- Good priority management of defects to ensure major bugs do not make it into the final release of the product.

Scope:

To ensure we do not get overwhelmed, we will not write too many tests at first. Instead, we will build up the test code as we go. Unit test all classes to ensure proper functionality but not go too overboard. Make sure to test not-functionals using either Visual Studio resources or outside resources.

Process Overview:

- Tests will be housed in a separate project in order to easily be able to remove them for the final release of the product. Utilize branching on GitHub to enable access for the whole team at once. Also to ensure that we have backups to revert to if something goes wrong.
- Create interfaces and fake classes for each class for quick and easy testing.
- Consistent style for testing both for readability and accessibility.

Process section:

- Requirements Testability

- Build tests to force requirements so that those requirements must be met.
- Document tests to show what they test and how thoroughly they test.
- Use test driven development for certain requirements when it is appropriate, such as ensuring read and writes are properly executed.
- Unit testing for classes will ensure that the code is stable and usable anywhere they are needed due to testing based on interfaces.
- Test Case Development
- Perform integration testing any time a new class is finished to ensure both the stability of the program as well as its functionality.
- Use requirements to create test cases to ensure that all requirements are met and to ease in the creating of tests.
- Once again, all classes will be unit tested (this can't be stated enough) to ensure that test cases are successful.
- Unit testing
- Use Visual Studio testing resources to ease in the creation of tests (more time for development).
- Unit testing will be done as we code to ensure that minimal defects slip through.
- Unit tests will be housed in a separate project in the solution to allow for easy removal for final project
- Ensure that unit tests are easy to use and helpful at finding defective code by indicating where the tests failed.
- System Testing
- Each member will spend time every week testing the system prior to merging files to ensure that the system as a whole doesn't break.
- We will research alternate testing methods to test non-functional requirements to ensure these are met.
- Defect Management
- Check into how to use Trac, to log, prioritize and resolve any defects.
- Update found defects ASAP (or as often as is reasonable) to ensure that they are known and therefore resolved quickly.

Post Iteration1 Update:

Unfortunately, due to some time constraints that we ran into, we were unable to reach a testing phase. A majority of our testing thus far has been comprised of simply running the program and ensuring that the features that we had implemented were reacting properly. This means that we have yet to stress test anything or do front end validation of input. Test development will be a high priority for the second Iteration.

Post Iteration 2 Update:

Great progress has been made in the QA department. Not as much as hoped for, but it is evolving well. We have arrived at a decision to use the NUnit test generator to create all of the tests, but without using the NUnit testing tools. This allowed us to create a series of files that basically

outline everything that needs to be done, only requiring a little tweaking in each file to fully implement the testing.

Unit testing for Data Types has been implemented for quite a few of the Models that we have (Many more to come). They are currently only done for the Client Side models, but implementing the service side tests for models is just a matter of copying the files and tweaking them. There have been a few hang ups, particularly with working with unfinished code. I have been hesitant to implement tests that check services and controllers since they are changing all the time. Once everything is a little more stable, work in this area will rapidly increase, especially when we begin testing as a team.

Post Iteration 3 Update (final):

During Iteration 3, we ran into some roadblocks that pushed out the development of sophisticated test services. However, after these roadblocks were removed, we were able to implement a testing API that allowed us to pick a service to test. This setup allowed us to implement a change to the services, select a service, and test the change. It is full featured, prompting you for parameters to pass in if it can accept parameters, and returning all of the data that the service would return in Json format.

With this new testing in place, we were able to carry out basic local variable testing through normal coding practices and then when we needed to ensure that a service was working properly, we could test it through the API. While this did not replace mocking the services, it allowed us to test the services very quickly and easily, which proved invaluable. What we have learned from this experience is that, moving forward, we need to learn more about how to implement testing and put that knowledge, plus what we have learned in this project, to good use.

Defect Report – Iteration 1

	3	View misalignment	There is an issue where edit boxes are staggered when shown in edit view specifically in Internet Explorer and Chrome.	MED	MED	Jordan
	4	Search Issue	When doing a search it also looks through the edit, delete, and details columns. Searching for ed, will not help filter results because of the edit column.	LOW	Low	Chase
	5	SSN issue	when adding a social security number to a user it displays as a float instead of a string of numbers with dashes	LOW	LOW	Chase
Fixed						
	2	Compilation errors	There is an issue with the current master where there are build errors even though the program runs fine	LOW	LOW	Chase
	1	Can't Edit Anything	Information from the edit view is unable to be sent to the Database	HIGH	HIGH	Everybody

Defect Report – Iteration 2

Open	ID	Title	Description	Severity	Priority	Owner
	3	View misalignment	There is an issue where edit boxes are staggered when shown in edit view specifically in Internet Explorer and Chrome.	MED	MED	Jordan
	4	Search Issue	When doing a search it also looks through the edit, delete, and details columns. Searching for ed, will not help filter results because of the edit column.	LOW	LOW	Chase
	6	EducationID issue	Looks like there may be a Difference in spelling between the model and the database when it comes to the educationId of Education class, for some reason when I built the tests it was trying educationID with a capitol D instead of lowercase which the class expects.	don't know	MED	John
	7	QuesitonId	In applicantQuestionAnswer, QuestionId is marked as QuesitonId....	MED	MED	John
Fixed						
	2	Compilation errors	There is an issue with the current master where there are build errors even though the program runs fine	LOW	LOW	Chase
	1	Can't Edit Anything	Information from the edit view is unable to be sent to the Database	HIGH	HIGH	Everybody
	8	OpenJob Details	Selecting a job opening to view details for returns a blank page	HIGH	HIGH	Patrick
	9	Return to OpenJobs	Clicking the "Return to Job Openings" link in a job opening's detail page causes an exception	HIGH	MED	Patrick

Source Code

Attached:

- AIM.zip
- AIM.Database Schema Script.sql