

Grad Apps 2.0 Design Documentation

Edward Vaisman Sadman Sakib Hasan

January 25, 2018

Revisions

Date	Revision	Description
22 November, 2017	1.0	Initial Document Draft
4 December, 2017	1.1	Included UML Class Diagram and Rationale
12 December, 2017	1.2	Updated the Class diagram (Model section Role separation), Added Entity-Model Diagram, Added Data descriptions
14 December, 2017	1.3	Updated Class diagram to match the new model, Updated Entity-Model Diagram, Edited Data descriptions
16 December, 2017	1.4	Added Interface Design and Traceability Matrix
17 December, 2017	1.5	Added Sequence Diagrams for the Component Design
19 January, 2018	1.6	Updated Data Dictionary and Model Diagram

Contents

1. Introduction	5
1.1. Purpose	5
1.2. Scope	5
1.3. Overview	6
2. System Overview	7
3. System Architecture	8
3.1. Architectural Design	8
3.2. Decomposition Description	9
3.3. Design Rationale	9
3.3.1. MVC	9
3.3.2. Roles	9
4. Data Design	11
4.1. Data Description	11
4.2. Data Dictionary	13
5. Component Design	15
5.1. Model	15
5.2. View	16
5.3. Controller	16
5.3.1. Sequence Diagrams	16
6. Human Interface Design	25
6.1. Overview of User Interface	25
6.2. Screen Images	25
6.3. Screen Objects and Actions	32
7. Requirements Matrix	33
8. Appendices	35
A. R-Descriptions	35

List of Figures

1. UML Class Diagram	8
--------------------------------	---

2.	Entity Relation Diagram	12
3.	Data Dictionary	14
4.	Sequence Diagram: Login to the system	17
5.	Sequence Diagram: Role Selection	18
6.	Sequence Diagram: Upload an Application	19
7.	Sequence Diagram: Update an Application	20
8.	Sequence Diagram: Assigning a Role	20
9.	Sequence Diagram: Applying filter	21
10.	Sequence Diagram: Notification	22
11.	Sequence Diagram: Review an Application	23
12.	Sequence Diagram: Applying University Assessment	24
13.	Login Page View	25
14.	Role Selection View	26
15.	Admin View - Committee Member Status	27
16.	Admin View - List Faculty Members	27
17.	Admin and Professor View - List of Student Applications	28
18.	Filters available for the Admin and Professor role	29
19.	Committee View - List of review applications	30
20.	Filters available for the Committee role	31

List of Tables

1.	Description table for the model functions	15
2.	Traceability Matrix for R-descriptions	34

1. Introduction

1.1. Purpose

This software design document is intended to provide information regarding the system and architecture on the design of the new graduate application system for the EECS graduate program.

1.2. Scope

Gradapps v2.0, a graduate application system, is intended to assist the EECS graduate program better locate and maintain potential graduate student applications from the moment it is submitted until the moment a decision is made. It is intended to replace the current system in use for the EECS graduate program. It will be accessible for use as an online portal on the web.

Gradapps v2.0 promises the following:

- *Administrators* will be able to manage *Committee Members*, assign applications to *Committee Members* to review, edit/manage student applications, grant/revoke privileges from faculty members and add/remove members from the system.
- *Committee Members* will be able to view and manage a list of student applications assigned to them and be able to save applications as a draft for future edits.
- *Professors* will be able to search for applications and indicate their interest in a student to other professors and for personal use indicate whether or not an application has been reviewed.
- All roles will be able to apply advanced filtering methods relative to their given roles to help search for applications.

1.3. Overview

This document is divided into 9 sections,

- Introduction
- System Overview
- System Architecture
- Data Design
- Component Design
- Human Interface Design
- Requirement Matrix
- Appendices

2. System Overview

Having a postgraduate degree on one's resume is always appealing to their future employer, let it be working in the industry or continuing studies to achieve a doctoral status. When applying for a postgraduate program, applicants spend a lot of their time gathering different levels of information (transcripts, letter of recommendation, resume etc.) required for admission. Once an application has been submitted, the graduate program analyses the information to find the best candidate for each program.

Analysing that level of dense information can be challenging at times and to avoid loss of any information, it is best practice to automate this process as much as possible. In order to achieve that goal, a *concise* and *simple* Business System is required that will reduce the manual work.

The graduate program in Electrical Engineering and Computer Science (EECS) is facing a very similar situation. The current system our client has involves a lot of manual work. The centre of this process are the Graduate Program Director (GPD) and Graduate Program Assistant (GPA) who plays a major role in all applications regardless of the applicant being admitted or rejected.

Administrators have the task of managing *Committee Members*, acquiring documents from the admissions office for upload, assigning applications to *Committee Members* for review, assigning roles to faculty members, uploading applications for faculty members to view, and lastly manage student applications.

Committee Members have the task of reviewing an application provided by the Administrator(s) and send it back for the *Administrators* to upload once all reviews are in for Faculty Members to see.

Professors then have the ability to search for applications that best suit their criteria for a good graduate student.

Given the current restrictions with York admissions office some things such as accessing different levels of information must be done manually by the *GPA* other types of accessing information such as, looking up a university description could be done automatically. Overall, our client requires a more robust and concise system that will enable them to automate the selection of the best candidate into the program minimizing the manual work.

3. System Architecture

3.1. Architectural Design

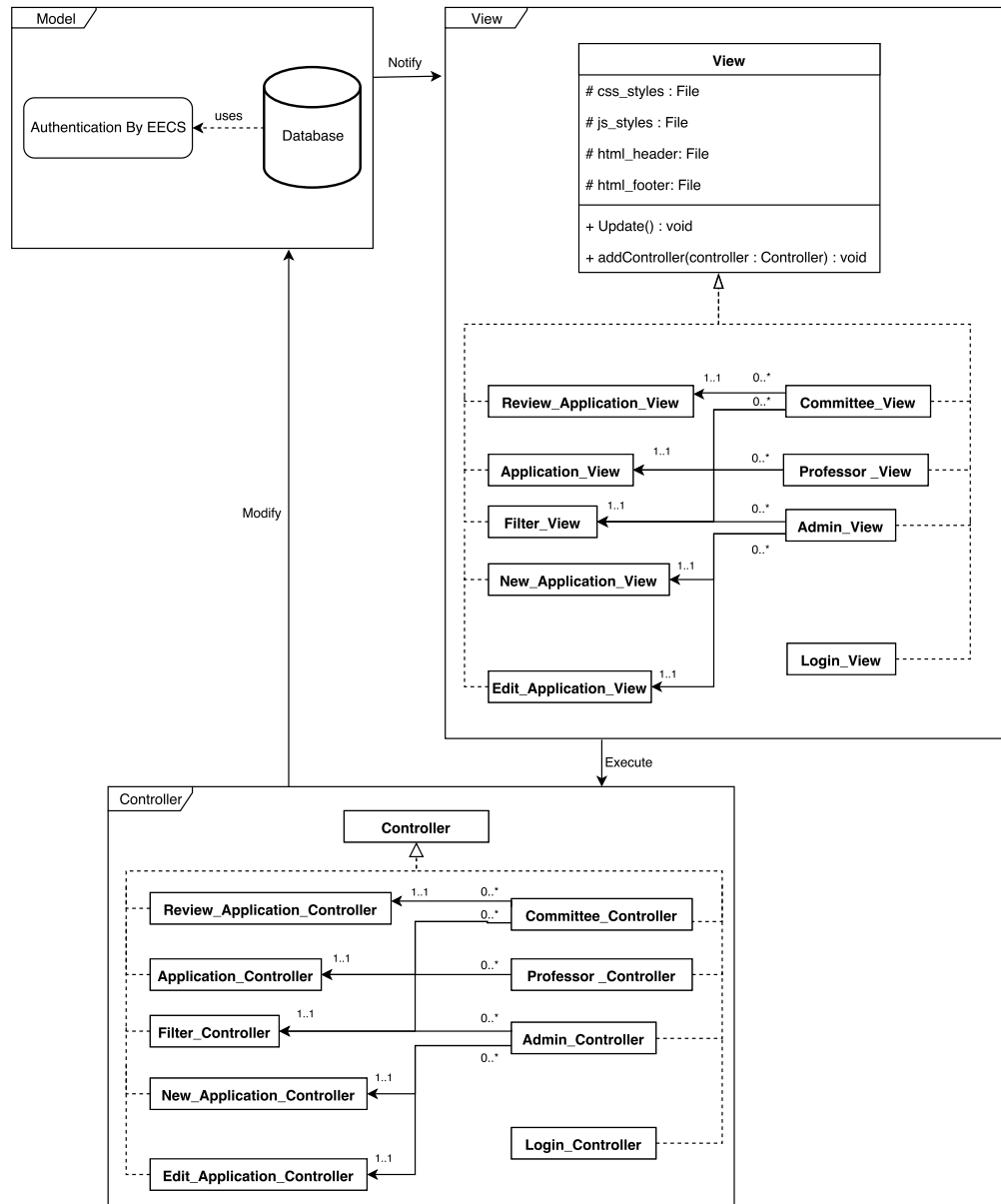


Figure 1: UML Class Diagram

3.2. Decomposition Description

The System Architecture was designed with the MVC Pattern in mind.

- The *Model* represents the database and acts as a data handler for all the data to be stored.
- The *View* represents the front-end web application using web technologies such as html, css and javascript.
- The *Controller* represents the back-end system that triggers the model and updates the view.

3.3. Design Rationale

3.3.1. MVC

Given that this is a web application designing it using the MVC pattern makes the most sense here for the following reason:

- Simultaneous development
- Ease of modification
- Multiple Views for our one model

As a result of MVC, developers will be able to work on parts of the system concurrently as opposed to having to wait for others to finish their part and the ability to add or remove a feature later on will be easier and the overall system will be modular.

3.3.2. Roles

To satisfy the specification requirements of faculty members being able to have more than one role, there is no inheritance/generalization between faculty members and all the other roles. This design choice allows for an easy separation of information across roles so that changes in one does not affect changes in another.

An alternative to this design would have been to split the roles into their own classes. This choice is not beneficial for our system since all of our roles share common things such as id, name, email etc and would create a lot of duplication in the system. See Section 4 for diagram and data description.

4. Data Design

4.1. Data Description

The following diagram illustrates the UML Entity Relationship. The design of each entity will be represented in the database during the implementation phase.

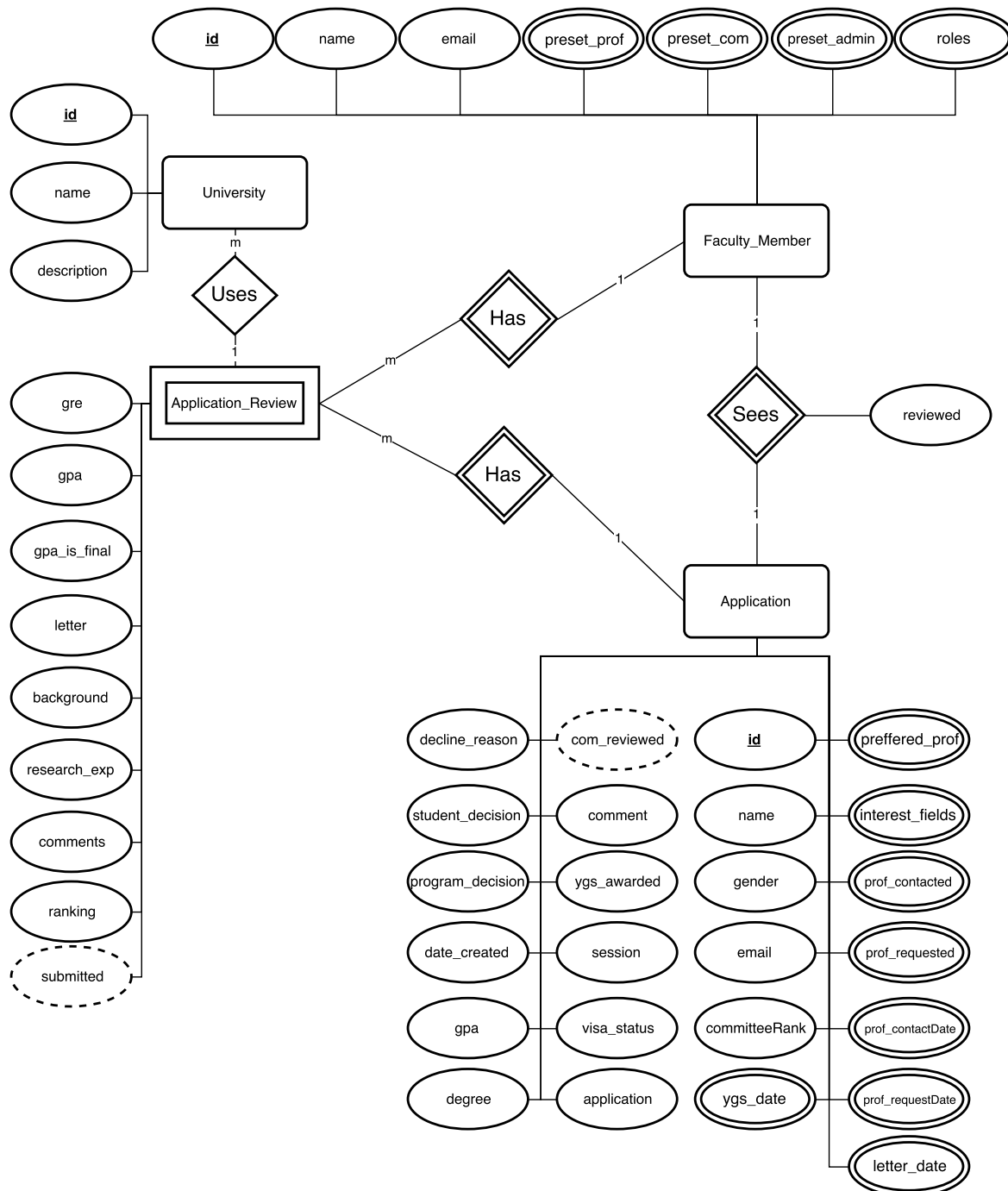


Figure 2: Entity Relation Diagram

4.2. Data Dictionary

Table Name	Field	Type	Null	Default
University	id	PK int	no	
	name	varchar(xx)	no	
	description	array of varchar(xx)	yes	NULL
Application_Review	com_id	FK, PK int	no	
	app_id	FK, PK int	no	
	gre	int	yes	NULL
	gpa	enum('A+', 'A', 'B+', 'B', 'C+', 'C', 'D+', 'D', 'E', 'F', 'NC')	yes	NULL
	gpa_is_final	bool	no	FALSE
	letter	varchar(xx)	yes	NULL
	background	varchar(xx)	yes	NULL
	research_exp	varchar(xx)	yes	NULL
	comments	varchar(xx)	yes	NULL
	ranking	enum('A+', 'A', 'B+', 'B', 'C+', 'C')	yes	NULL
	submitted	bool	no	FALSE
	last_reminded	Date	no	current time
	date_assigned	Date	no	current time
Application	id	PK int	no	
	name	varchar(xx)	no	
	gender	enum('male', 'female')	yes	NULL
	email	varchar(xx)	no	
	gpa	enum('A+', 'A', 'B+', 'B', 'C+', 'C', 'D+', 'D', 'E', 'F', 'NC')	yes	NULL
	committeeRank	enum('A+', 'A', 'B+', 'B', 'C+', 'C')	no	
	degree	enum('PHD', 'Msc', 'MASC')	yes	NULL
	visa_status	enum('Domestic', 'Visa')	no	
	application	File	no	
	session	enum('Fall', 'Winter', 'Summer')	no	
	interest_fields	array of varchar(xx)	yes	NULL
	preferred_prof	array of varchar(xx)	yes	NULL
	date_created	Date	no	current time
	prof_contacted	array of varchar(xx)	yes	NULL
Application	prof_requested	array of varchar(xx)	yes	NULL
	prof_contactDate	array of Date	yes	NULL
	prof_requestDate	array of Date	yes	NULL
	letter_date	array of Date	yes	NULL
	ygs_date	array of Date	yes	NULL
	program_decision	varchar(xx)	yes	NULL
	student_decision	varchar(xx)	yes	NULL
	decline_reason	varchar(xx)	yes	NULL
	ygs_awarded	bool	no	FALSE
	comments	varchar(xx)	yes	NULL
	com_reviewed	bool	no	FALSE

Figure 3: Data Dictionary Part.1

Table Name	Field	Type	Null	Default
Application_Seen	prof_id	FK, PK int	no	
	app_id	FK, PK int	no	
	reviewed	bool	no	FALSE
Faculty_Member	id	PK int	no	
	name	varchar(xx)	no	
	email	varchar(xx)	no	
	preset_prof	array of varchar(xx)	yes	NULL
	preset_com	array of varchar(xx)	yes	NULL
	preset_admin	array of varchar(xx)	yes	NULL
	roles	array of enum('Admin','Committee','Professor')	yes	NULL

Figure 4: Data Dictionary Part.2

5. Component Design

5.1. Model

The model acts as a data storage handler for the GradApps business system. Most of which is already discussed in Section 4. The database will use MySQL version: 5.7.20 and Authentication By EECS as per the requirements elicited from the client. The design chosen was in favor of simplicity instead of efficiency. This decision was based on the relatively low amount of student applications gradapps will store (roughly 1000 applications per year) thus eliminating the need for faster query calls.

5.2. View

Some views will be able to display other views depending on the role selected.

Here are a few:

- Admin View includes the Filter View, New Application View, Edit Application View, and the Application View.
- Committee View includes the Filter View, and the Review Application View.
- Professor View includes the Application View, and the Filter View.

Based on what the main view is (Admin, Committee, Professor) the filter view will adjust accordingly to allow for their respective controllers.

5.3. Controller

Since some roles have different privileges than others, it is important to restrict what one user could do based on the view that they are in. For example, if someone is in the professor view they cannot access methods from the admin controller.

5.3.1. Sequence Diagrams

The following sequence diagrams describe the communication between our actors and the system under description. In short our system behaves as follows.

- Actor acts on the system
- Client side validation
- Appropriate updates to database

- Return updated results and error message if necessary.

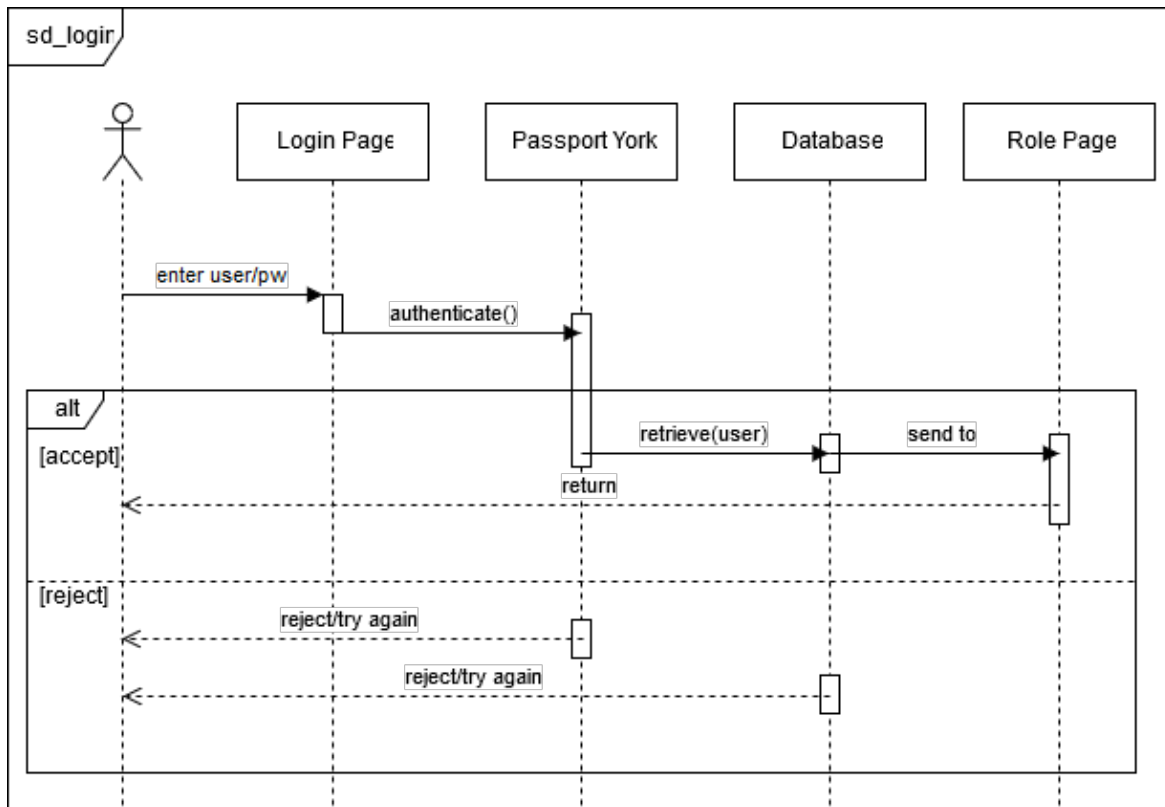


Figure 5: Sequence Diagram: Login to the system

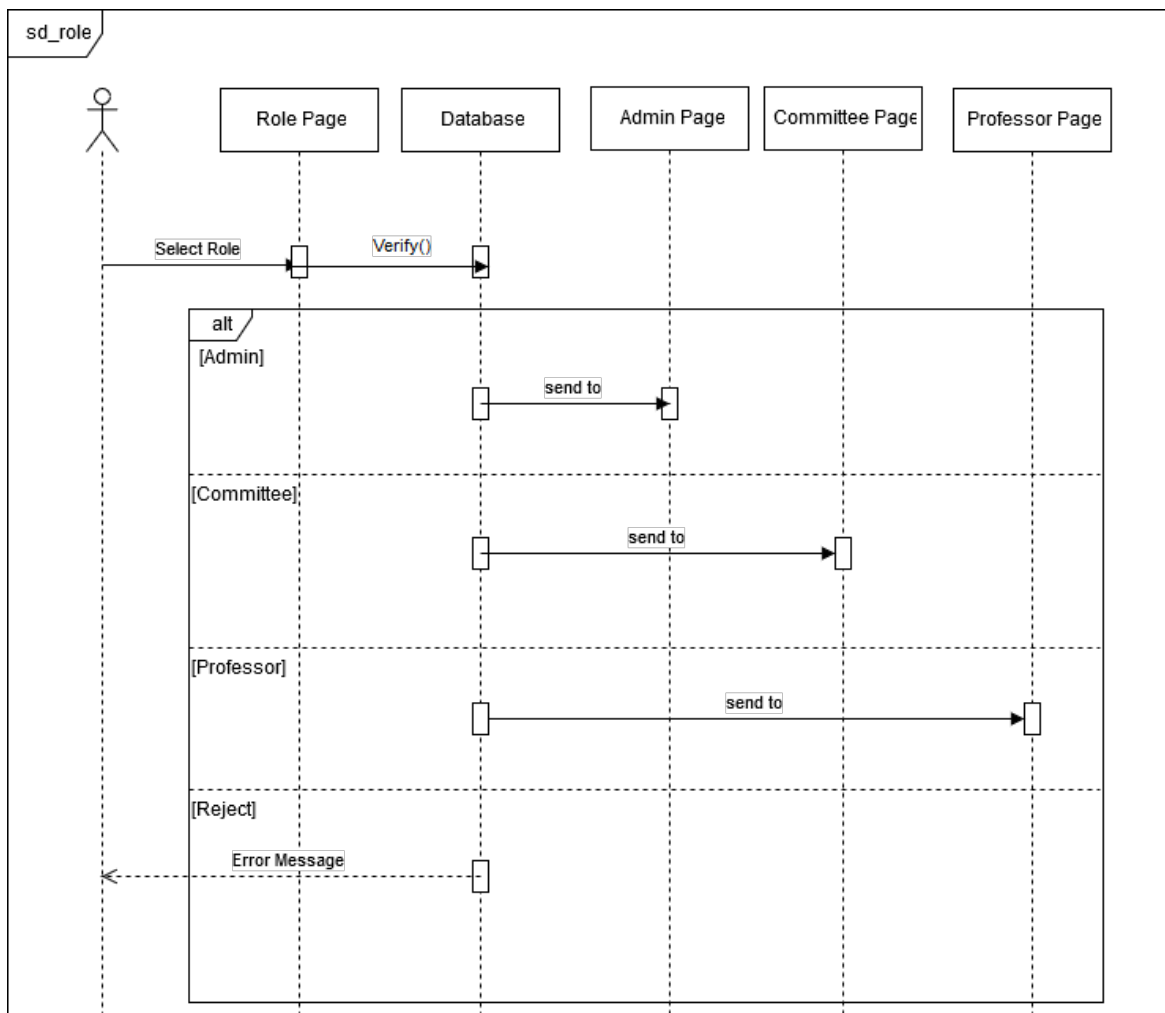


Figure 6: Sequence Diagram: Role Selection

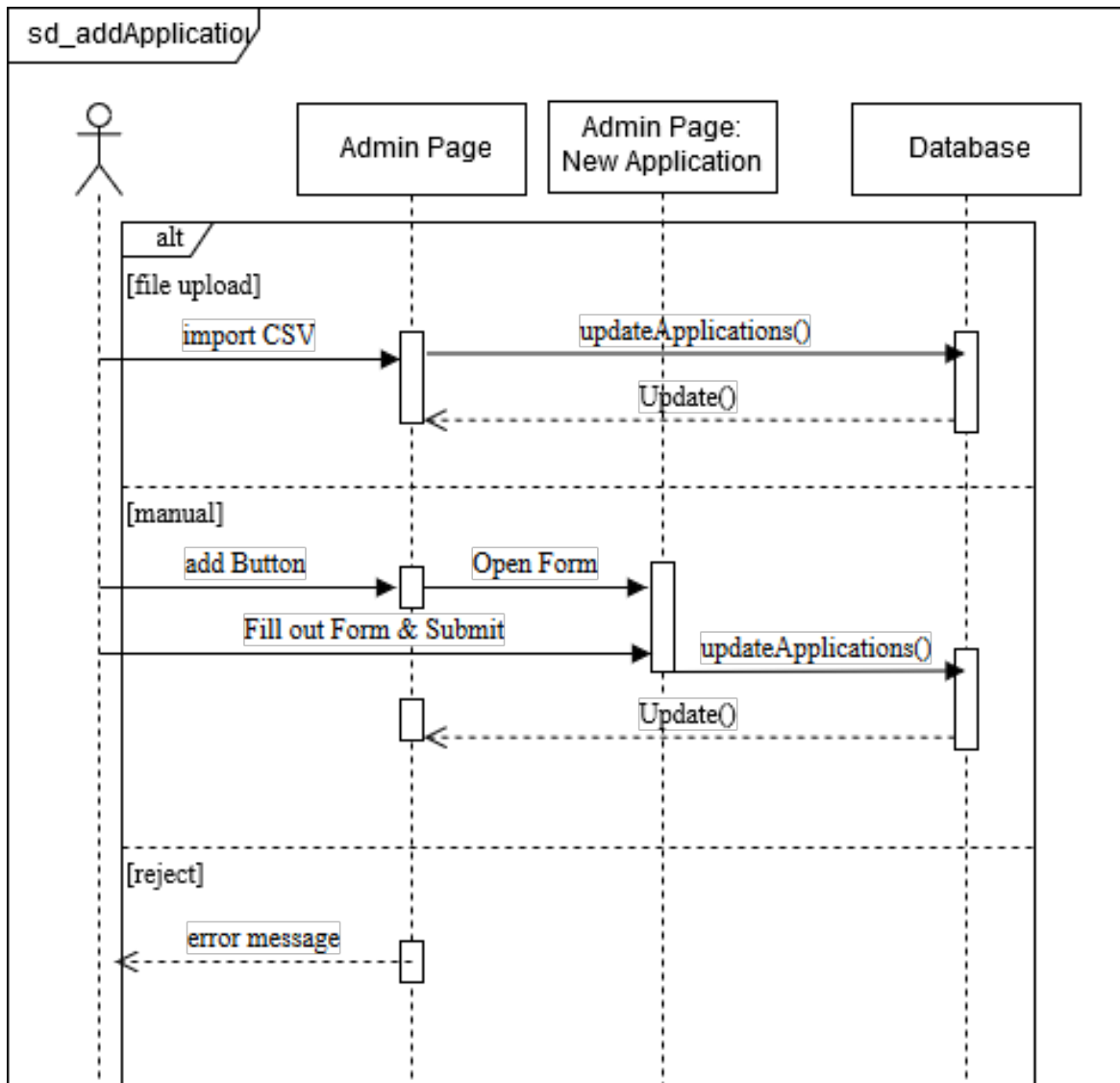


Figure 7: Sequence Diagram: Upload an Application

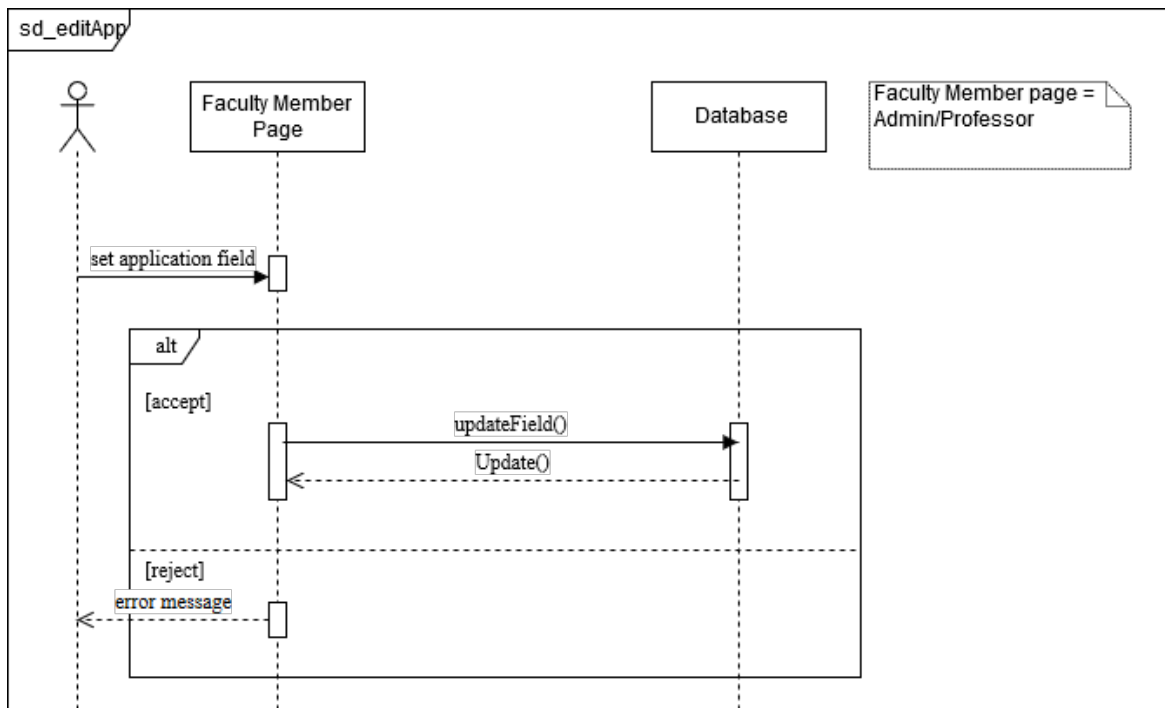


Figure 8: Sequence Diagram: Update an Application

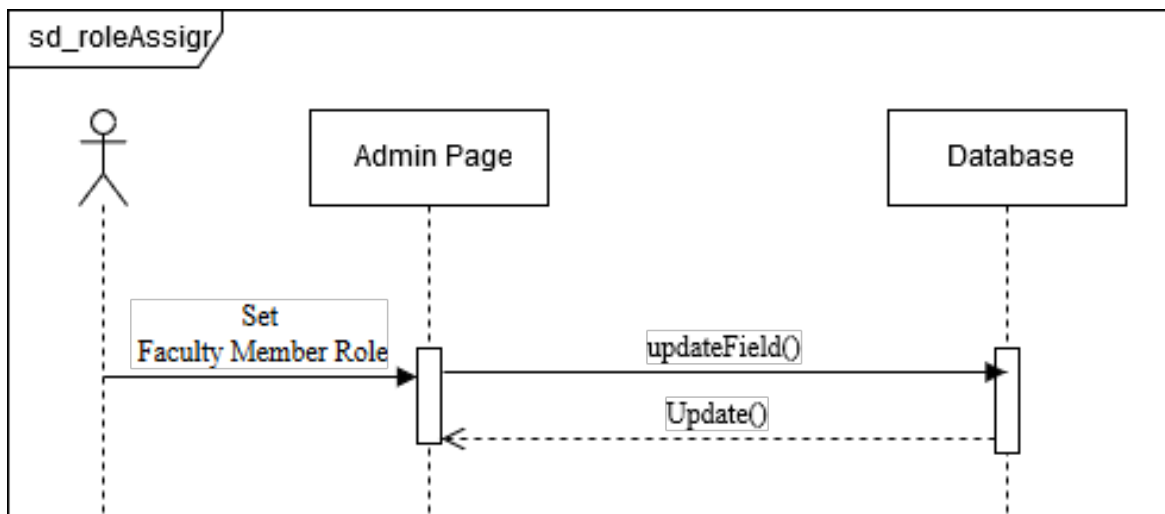


Figure 9: Sequence Diagram: Assigning a Role

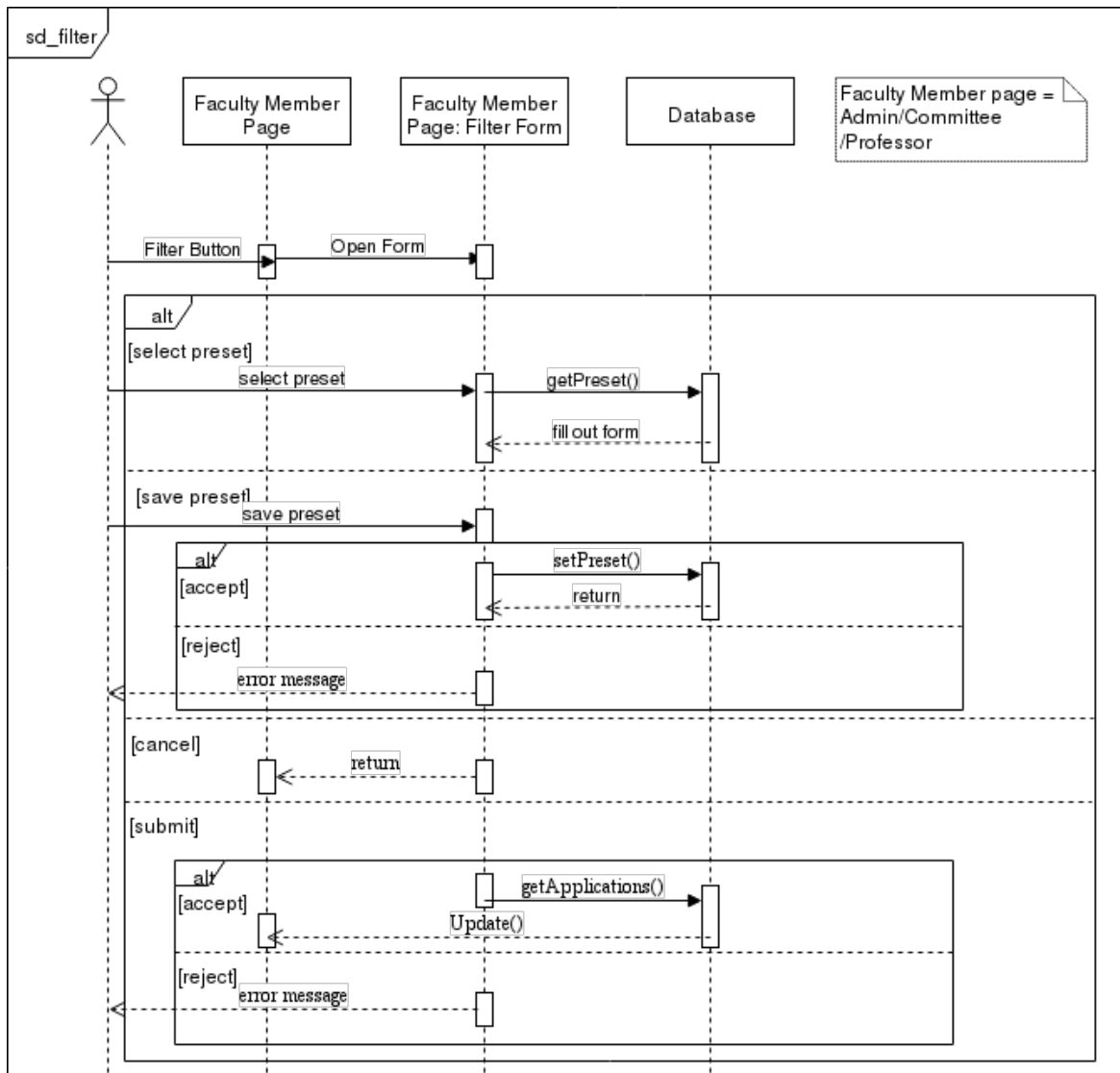


Figure 10: Sequence Diagram: Applying filter

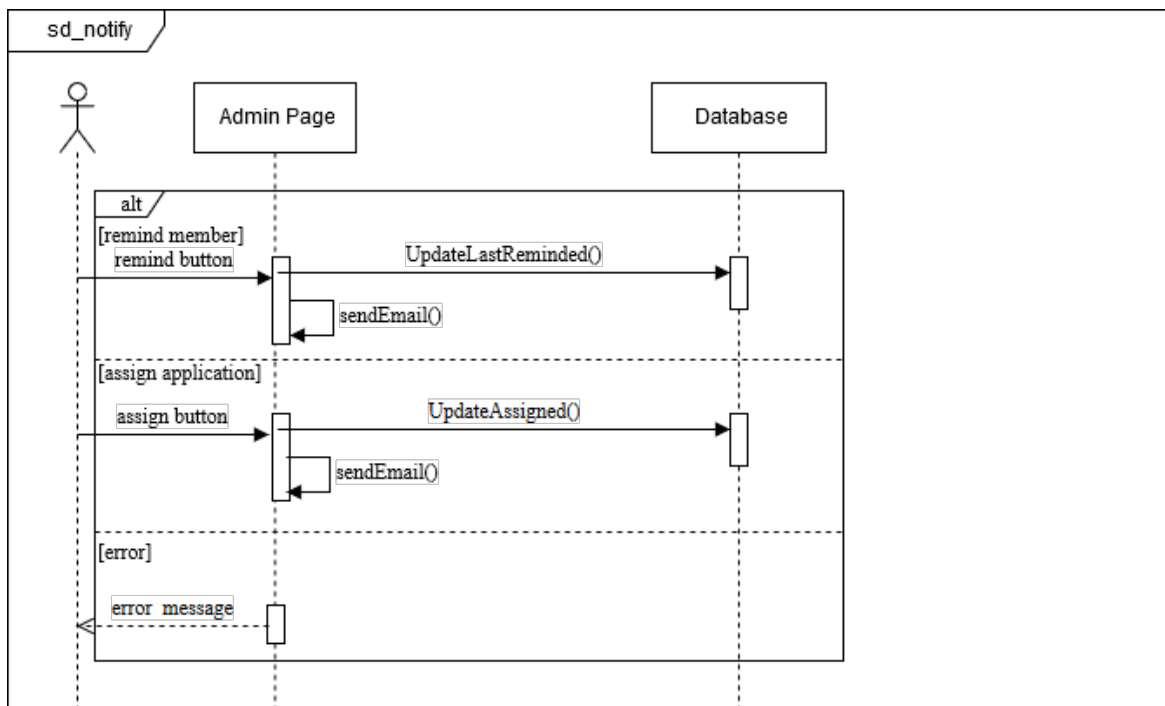


Figure 11: Sequence Diagram: Notification

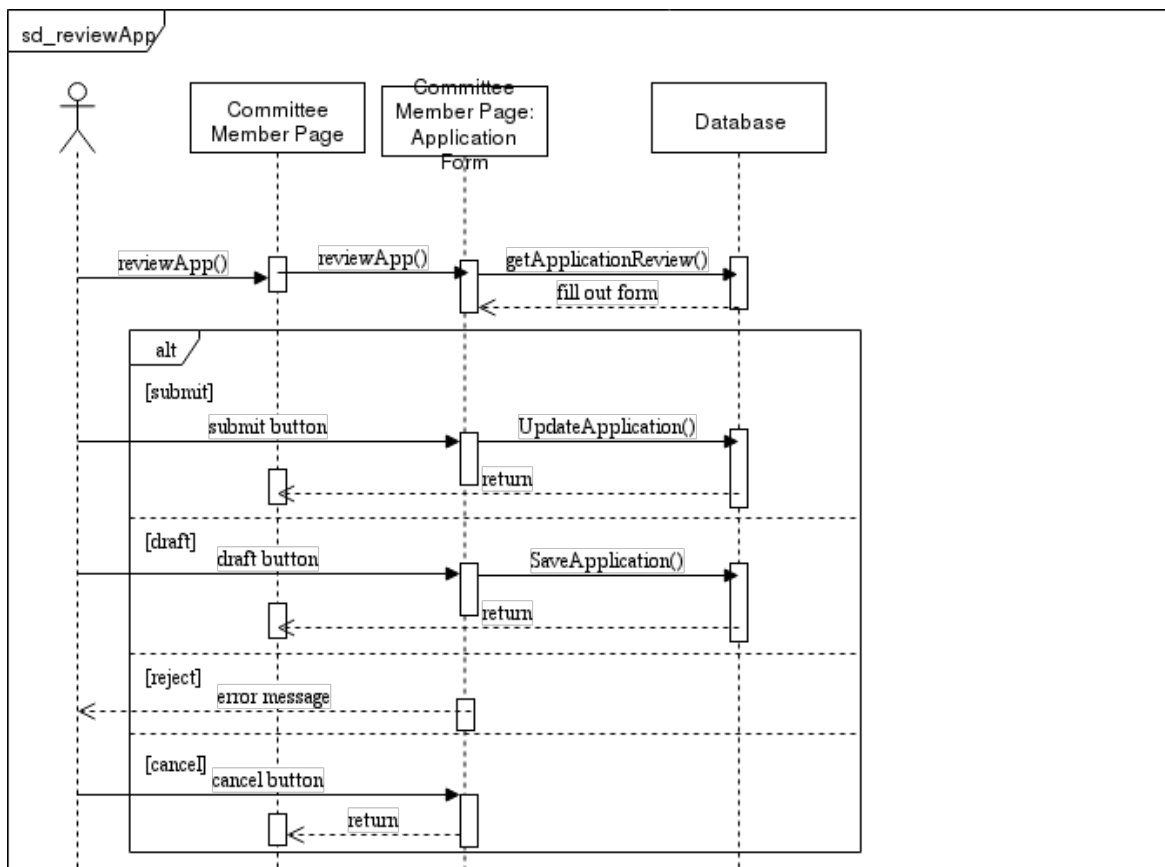


Figure 12: Sequence Diagram: Review an Application

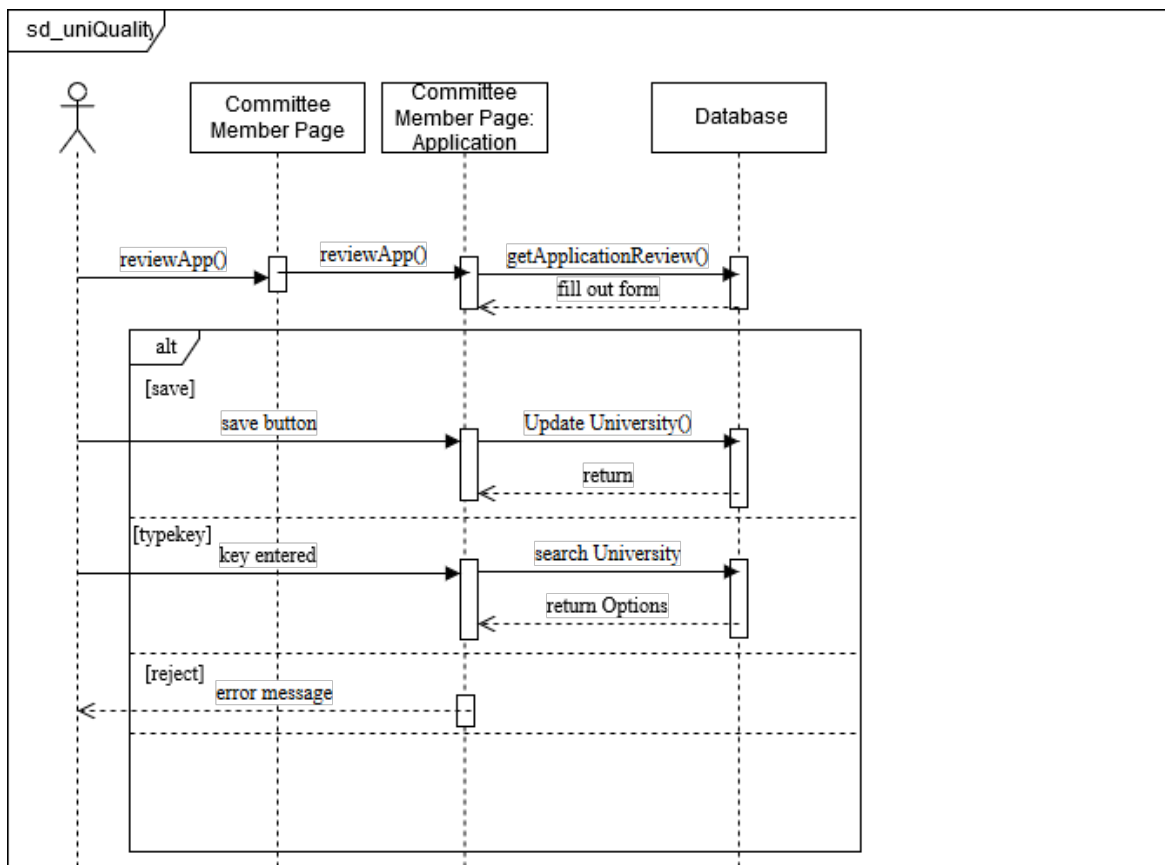


Figure 13: Sequence Diagram: Applying University Assessment

6. Human Interface Design

6.1. Overview of User Interface

Refer to <http://www.cse.yorku.ca/~eddyv/gradapps/index.html>

6.2. Screen Images

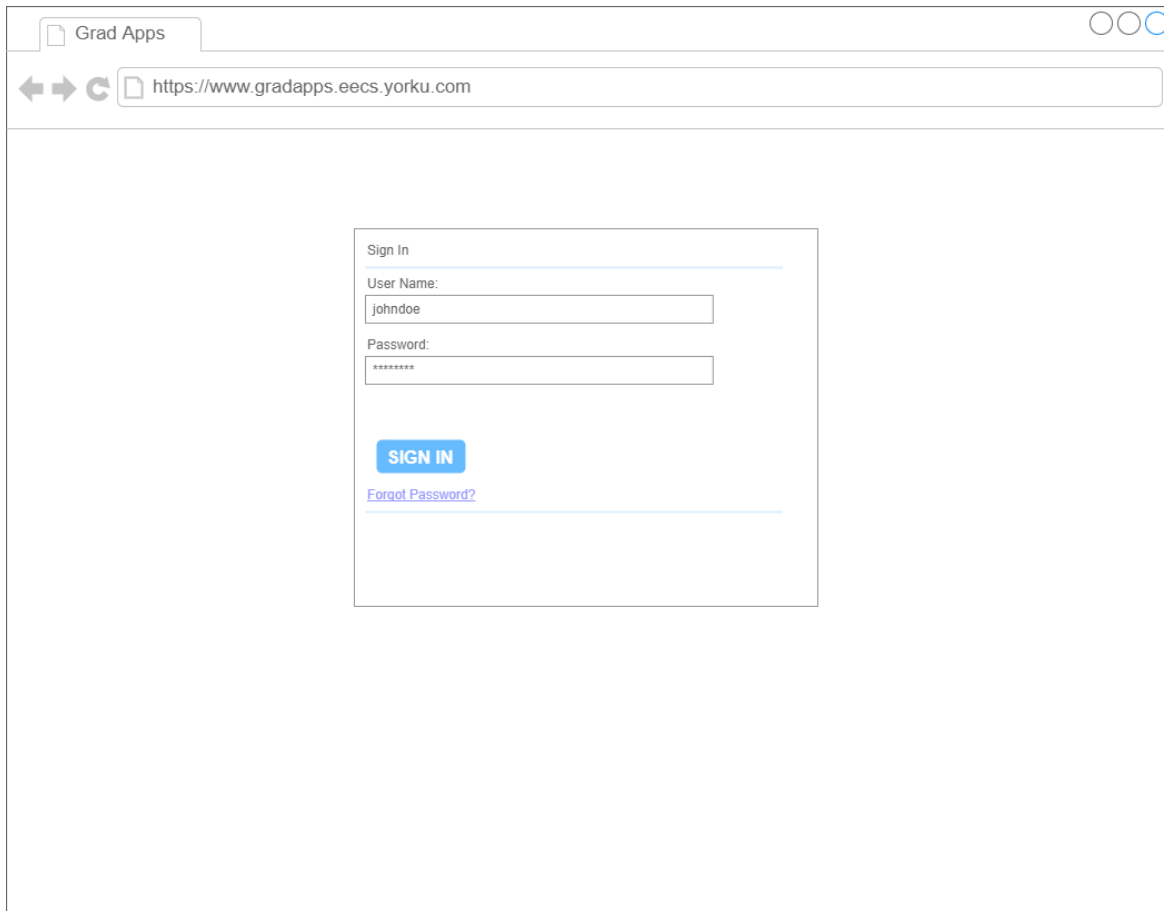


Figure 14: Login Page View

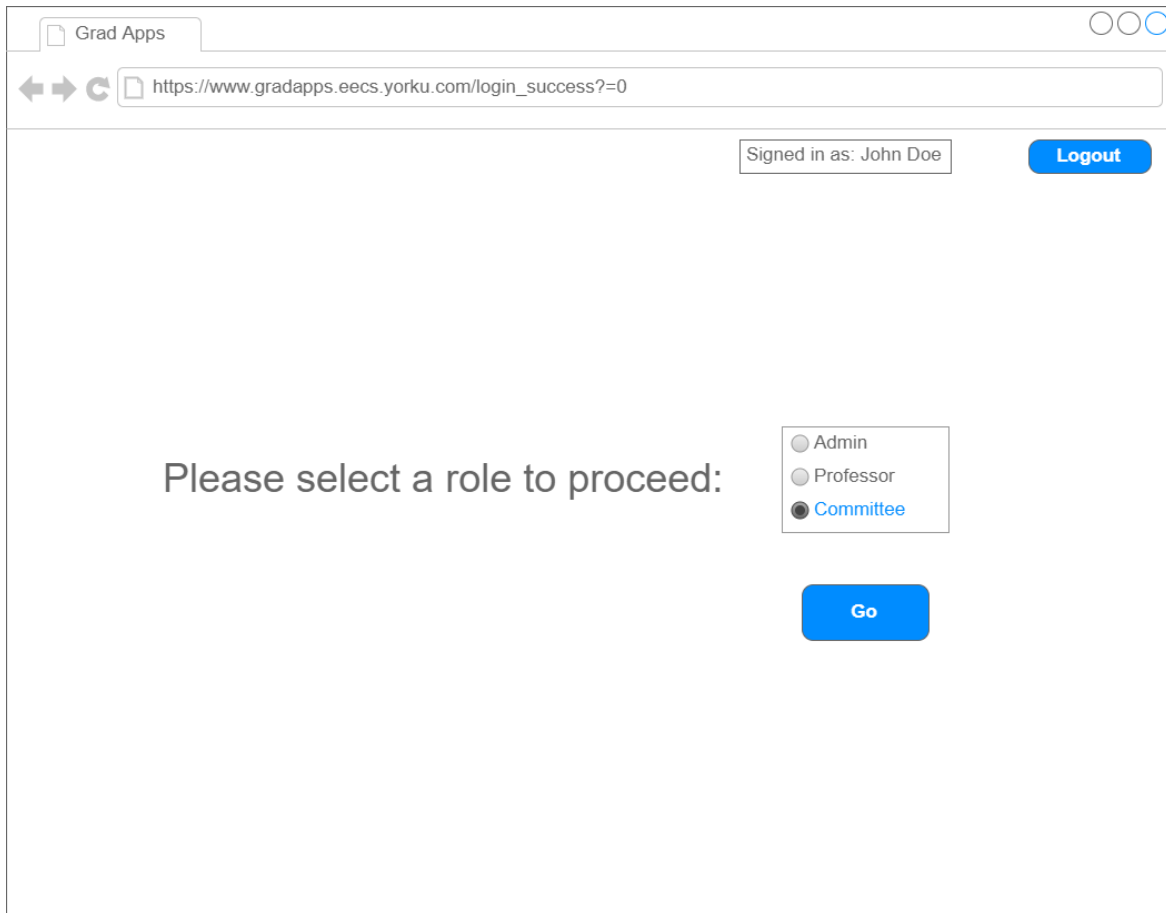


Figure 15: Role Selection View



<div>  Signed in as: Professor Awesome <div>Filter </div> <div>Role: Admin ▾</div> <div>Sign Out</div> </div>			
Committee Member Status	Faculty Members	Applications	
Name	Total Applications	# of Applications to Review	Actions
Michael R. M. Jenkin	10	6	<button>Remind</button>
Franck van Breugel	5	2	<button>Remind</button>
Andrew Eckford	3	1	<button>Remind</button>
Jonathan S. Ostroff	7	3	<button>Remind</button>
Melanie Baljko	5	2	<button>Remind</button>
Matthew Kyan	6	0	<button>Remind</button>

Figure 16: Admin View - Committee Member Status





<div>  Signed in as: Professor Awesome <div>Filter </div> <div>Role: Admin ▾</div> <div>Sign Out</div> </div>		
Committee Member Status	Faculty Members	Applications
Name	Role(s)	Action
Michael R. M. Jenkin	Professor	<button>Edit Role</button>
Franck van Breugel	Admin, Professor, Committee	<button>Edit Role</button>
Andrew Eckford	Professor	<button>Edit Role</button>
Jonathan S. Ostroff	Professor	<button>Edit Role</button>
Melanie Baljko	Committee	<button>Edit Role</button>
Matthew Kyan	Professor	<button>Edit Role</button>
Ourma	Admin	<button>Edit Role</button>

Figure 17: Admin View - List Faculty Members

EECS
ELECTRICAL ENGINEERING
AND COMPUTER SCIENCE

Signed in as: Professor Awesome

Filter 

Role: Admin 

Sign Out

Committee Member Status

Faculty Members

Applications

of applications with no reviewers: 3

Create New Application

Field of Interest	Committee Ranking	Degree Applied For	Visa Status ↓	Application Status	Actions
AI	A+	PHD (CS)	CAD	Contacted by Michael R. M. Jenkin	<div>View Application</div> <div>Set to Contacted</div> <div>Set to Reviewed</div>
Theory of Computation, Networks, AI	A+	MSc	CAD	Accepted	<div>View Application</div> <div>Set to Contacted</div> <div>Set to Reviewed</div>
Biomedical Engineering, Software Engineering, VR	C	MASc	CAD	Rejected	<div>View Application</div> <div>Set to Contacted</div> <div>Set to Reviewed</div>
Theory of Computation, Networks	A+	PHD (CS)	VISA	Submitted	<div>View Application</div> <div>Set to Contacted</div> <div>Set to Reviewed</div>

Figure 18: Admin and Professor View - List of Student Applications

Filter

Choose Your Columns

GPA

Degree Applied For

Visa Status

Application Status

Reviewed Status

Selected Columns

Field of Interest ✕

Preferred Professor(s) ✕

Name ✕

Committee Ranking ✕

Choose Your Filters

Name ▾

Field of Interest ▾

Preferred Professor(s) ▾

Committee Ranking ▾

GPA ▾

Degree Applied For ▾

Visa Status ▾

Application Status ▾

Reviewed Status ▾



Selected Filter

Visa Status = VISA AND GPA >= C AND Committee Ranking > C

✕ Cancel

✔ Submit

Figure 19: Filters available for the Admin and Professor role

 **EECS** ENGINEERING, ELECTRONICS, AND COMPUTER SCIENCE Signed in as: Professor Awesome Filter  Role: Committee ▾ Sign Out

Example Tables

Date of application	Student #	Status of application	Actions
31/12/2017	1	Reviewed	View Application
25/10/2017	2	Not Reviewed	Edit Application
20/10/2017	3	Draft Available	Edit Application
03/09/2017	4	Reviewed	View Application
04/04/2017	5	Not Reviewed	Edit Application
1/1/2017	6	Reviewed	View Application

Figure 20: Committee View - List of review applications

Filter

Choose Your Columns

Preset 1 ▼

Student # Application Status

Selected Columns

Date of application ✕

Choose Your Filters

Date of application - Application Status -

Selected Filter

Field of Interest = AI AND Preferred Professor = Michael R. M. Jenkin
AND Status of application = Not Reviewed



✕ Cancel  Submit  Save Filter

Figure 21: Filters available for the Committee role

6.3. Screen Objects and Actions

The screenshots un the previous section from Figure 13 to 20 describes the flow of action an user may take for the System Under Description. The following describes each figure from the previous section.

- Figure 13 specifies the login view for the system. The login credentials shall be authenticated by Passport York. The user is expected to have a valid Passport York account and be part of the system database in order to gain access to the system.
- Figure 14 specifies the role selection view upon logging into the system. Once the user has been successfully authenticated into the system, the user needs to select a role to proceed further.
- Figure 15 specifies the admin view of checking the committee member status. This view includes reminding a committee member who have been assigned a review and has not completed it yet.
- Figure 16 specifies the admin view that lists all the members in the system. An admin can remove or change a role of a member. An admin can further add a new member into the system.
- Figure 17 specifies the admin and professor view of all student applications in the system.
- Figure 18 specifies the admin and professor view of the filters available on student applications.
- Figure 19 specifies the committee member view of all applications needed for review in the system.
- Figure 20 specifies the committee member view of the filters available on review applications.

7. Requirements Matrix

The following table traces the primary R-descriptions from the Requirement Documentation which satisfies the design specified from Figure 13 to 20. The R-descriptions are included in Appendix A to make it convenient for the reader.

REQ	Fig. 13	Fig. 14	Fig. 15	Fig. 16	Fig. 17	Fig. 18	Fig. 19	Fig. 20
REQ1	X							
REQ2	X	X	X	X	X	X	X	X
REQ3	X	X	X	X	X	X	X	X
REQ4		X	X	X	X	X	X	X
REQ5		X						
REQ6		X	X	X	X	X	X	X
REQ7				X				
REQ8				X				
REQ9				X				
REQ10			X					
REQ11					X			
REQ12					X			
REQ13					X			
REQ14			X					
REQ15					X			
REQ16							X	
REQ17							X	
REQ18								X
REQ19							X	
REQ20							X	
REQ21					X			
REQ22					X			
REQ23					X			
REQ24					X			
REQ25						X		
REQ26							X	
REQ27				X				
REQ28							X	

Table 1: Traceability Matrix for R-descriptions

8. Appendices

A. E/R-Descriptions

A.1. Use Case 1 Authentication - E/R Descriptions

ENV1	Each EECS graduate program member must have an unique username and password associated with their account via PPY.
------	--

Rationale: The login credentials will be handled through the Passport York. This is to simplify login uses and since this business system is under York University, PPY is the most secure option.

ENV2	If the PPY is down for maintenance service, the system must not be accessible.	See Env. ??
------	--	-------------

Rationale: Since the system heavily relies on PPY authentication, if PPY is scheduled down for maintenance, the system becomes inaccessible.

REQ1	A user shall be able to login to the system <i>iff</i> they are a member of the EECS graduate program with a role assigned.	$user \in FM$
------	---	---------------

Rationale: A user shall be able to log into the system if they are a member of the EECS graduate program and have a role assigned to them. The login will be completed using Passport York Authentication (PPY).

REQ2	A user shall be able to logout of the system <i>iff</i> they are already logged in.
------	---

Rationale: A user shall be able to log out of the system iff they are logged into the system through PPY.

REQ3	A user shall be not both logged in and logged out at the same time.	$(m_loggedIn) \iff \neg(m_loggedOut)$
------	---	---

Rationale: For security reasons a user cannot be both logged in and logged out at the same time. If a logged in account gets idle after 15 minutes, the user is automatically logged out (refer to Req. 4).

REQ4	A user shall be logged out of the system after a maximum of 15 minutes of idleness.
------	---

Rationale: It is best practice to log out an idle user from a business critical system. This is to make sure the user does not stay logged on forever and to ensure liveness property on a user account.

REQ5	A user shall select a role from the list of roles they are assigned to once logged into the system.	List of all the roles are as specified in Section ??.
------	---	---

Rationale: The roles available to the user upon logging into the system are the roles the user have been assigned to. A user cannot select a role that has not been assigned to them.

REQ6	A user shall be logged in with exactly <i>one</i> role at any given time.	$\forall role1, role2 : (user.role = role1 \wedge user.role = role2) \implies (role1 = role2)$
------	---	--

Rationale: A logged in user cannot be logged in with two different roles. This is to avoid conflicting access control between two different roles.

A.2. Use Case 2 Add Student Application- E/R Descriptions

ENV3	The <i>GPA</i> must manually compile an application received from the graduate office before sending it for review.
------	---

Rationale: This is the manual work that needs to be done by the Graduate Program Assistant. The GPA has agreed it is in their best interest to manually compile all bits of information into one file before proceeding with the application.

REQ7	An <i>admin</i> shall be able to upload a student application to the portal.
------	--

Rationale: Only an *admin* user shall be able to assign applications for review to a graduate committee member. For simplicity, there is no maximum cap enforced for reviewing an application by a committee member.

A.3. Use Case 3 Add/Remove Members and Assign Roles- E/R Descriptions

REQ8	An <i>admin</i> shall be able to add a new member to the list of EECS graduate program staff.
------	---

Rationale: Only an *admin* user shall be able to add a new member to the list of EECS graduate program staffs and assign one or more role(s) to them. In fact, an *admin* shall be able to add a new member and assign them to the role of *admin* as well.

REQ9	An <i>admin</i> shall be able to remove a member from the list of EECS graduate program staff except themselves.
------	--

Rationale: Only an *admin* user shall be able to remove an existing member from the list of EECS graduate program staffs. An *admin* **cannot** remove themselves from the system.

REQ10	An <i>admin</i> shall be able to assign a new role to an existing faculty member with an old role except for themselves.
-------	--

Rationale: Only an *admin* user shall be able to change existing roles of a registered user. An *admin* **cannot** change their own role in the system.

REQ11	An <i>admin</i> shall be able to remove a role from an existing graduate program member with a at least one role assigned.
-------	--

Rationale: A *admin* shall be able to remove a role assigned to an existing graduate member. This is to take away privileges on performing certain actions in the system.

A.4. Use Case 4 Assign Application to Committee Member - E/R Descriptions

REQ12	An <i>admin</i> shall be able to assign applications for review to a graduate committee member.	The set GC denotes the set of all graduate committee members which is a subset of all members in the system, $GC \subseteq FM$.
-------	---	--

Rationale: Only an *admin* user shall be able to assign applications for review to a graduate committee member. For simplicity, there is no maximum cap enforced for reviewing an application by a committee member.

REQ13	A <i>committee member</i> shall be notified when a batch of applications come in for review.
-------	--

Rationale: After an *admin* sends out application(s) for review to a committee member, the *committee member* gets an in app notification associated with their PPY.

A.5. Use Case 5 View/Filter Applications - E/R Descriptions

REQ14	An <i>admin</i> shall be able to export the applications to CSV format.
-------	---

Rationale: Only an *admin* user shall be able to download a CSV format of the applications and make changes to the file. Being able to import the changed CSV file is an optional requirement that may be implemented if time permits and all required deliverables have been achieved.

REQ15	A <i>committee member</i> shall be able to see the list of assigned application(s).
-------	---

Rationale: A *committee member* shall be allowed to view a list of new (to be reviewed) and previously reviewed application(s). This will allow the committee members to have an organized view of the applications left to review and the applications that have been reviewed already.

REQ16	A <i>committee member</i> shall be able to apply filtering only on selected attributes.	Refer to Table ??
-------	---	-------------------

Rationale: Allowing extra filtering on assigned application will cause favouritism of some student X while student Y will never get reviewed. Thus, it has been decided to only allow filtering on already reviewed application(s).

REQ17	An <i>admin</i> shall be able to view a list of all student application(s).	refer to Req. 19
-------	---	------------------

REQ18	An <i>professor</i> shall be able to view a list of student application(s) approved by an admin.	refer to Req. 19
-------	--	------------------

Rationale: An *admin* and *professor* shall be allowed to view a list of student application(s). The list of students shown can be narrowed/expanded using filters (refer to Req. 19).

REQ19	An <i>admin</i> and <i>professor</i> shall be able to filter on all student applications including personal attributes such as review status.
-------	---

Rationale: An *admin* and a *professor* shall be able to apply filters on a list of student applications. This is to allow narrow/expand searches of student on various attributes.

A.6. Use Case 6 Update Application- E/R Descriptions

ENV4	The <i>GPA</i> must contact the institutions graduate admission office whenever there needs to be a form of communication.
------	--

Rationale: The GPA can contact the institutions graduate admission office for further information on application if needed.

ENV5	The <i>GPA</i> must send the final decision of a student application to the institutions graduate admission office.
------	---

Rationale: The GPA must send the final decision of an application to the institutions graduate admission office.

REQ20	An <i>admin</i> shall be able to update all attributes of a student application.	refer to Table ??
-------	--	-------------------

Rationale: Only an *admin* user shall be able to update all attributes of an application (refer to Table ??). This to allow updating secured information on the application that maybe confidential.

REQ21	The <i>GPA</i> shall be notified when a <i>professor</i> has requested a student for admission.	refer to Env. ??
-------	---	------------------

Rationale: The *GPA* shall be notified when a *professor* requests a student for admission. The graduate program assistant can then contact the graduate office with further information (refer to Env. ??).

REQ22	A <i>professor</i> shall be able to review a student application.
-------	---

Rationale: A *professor* shall be allowed to view a posted student application. Once an application has been viewed by the professor, it is automatically checked **Reviewed** and this status is only visible to the professor user who have viewed it. For all not reviewed applications, the **Reviewed Status** field is left empty.

REQ23	A <i>professor</i> shall be able to contact a student.
-------	--

Rationale: A *professor* shall be allowed to contact a student once their application satisfies the professor's need. The contact shall need to be made through external email and the professor needs to update the application to **Contacted**. We leave this upto the user to use this feature correctly. The contacted status is visible to all graduate program members who have access to the portal (i.e all *admins* and *professors*).

REQ24	A <i>professor</i> shall be able to re-request a student for admission.	refer to Req. 21 and Env. ??
-------	---	------------------------------

Rationale: A *professor* shall be allowed to request a student for admission. Once the request has been sent in, the *GPA* will receive a notification (refer to Req. 21) and then can reach out the institutions graduate office with a request for admission (refer to Env. ??).

A.7. Use Case 7 Review Student Application- E/R Descriptions

REQ25	The <i>GPA</i> shall be notified when reviews for an application have been completed.
-------	---

Rationale: Only the *GPA* shall be notified when the review has been completed. It shall be a 1-to-1 relation to avoid multiple levels of notification between users.

REQ26	A <i>committee member</i> shall be able to save ongoing reviews as a draft for future completion.
-------	---

Rationale: A *committee member* shall be allowed to save ongoing reviews as a draft. A draft can be later resumed for completion. This gives the committee member an opportunity to not rush through the review.

REQ27	A <i>committee member</i> shall be able to view, use, add or modify a university assessment if it has already been provided.
-------	--

Rationale: A *committee member* shall be allowed to view, use or modify a previous university assessment used in a review. This removes the work of typing in the same university assessment or even a modified assessment of a previously assessed university.

REQ28	A <i>committee member</i> shall be able to submit a completed review to the <i>GPA</i> .	Refer to Req. 25
-------	--	------------------

Rationale: A *committee member* shall be able to submit a completed review to the *GPA*. The *GPA* can then upload the application to the portal for professors to access the application.