System Analysis & Design Project:
School Management System

Contents

Introduction

This project was created with the combined efforts of Kristian Kesar, Thomas Radulescu, and Filip Raguž for the Systems Analysis and Testing class of the academic year 2024/2025.

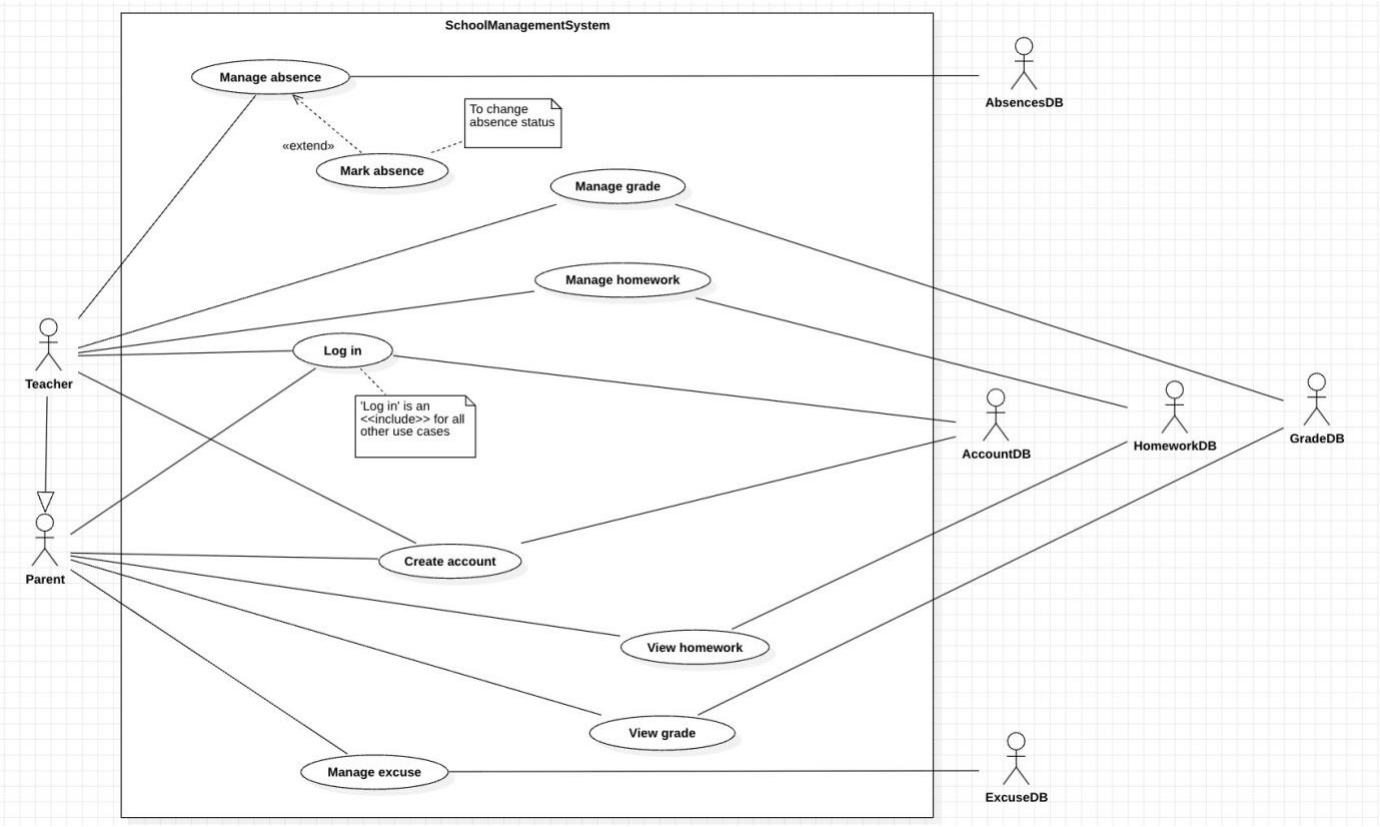
Summary

A primary school (up to 4th grade) management system for teachers and parents to easily track students' academic progress. To use the system, teachers and parents are required to create an account and use it to log in. Although students are a part of the system, they are not the intended users. It is assumed that one teacher manages one class throughout a school year.
The system will display a different menu depending on the type of user that is logging in.
For teachers, it will show a list of students in their class and an option to add homework. They can monitor any homework by adding due dates and marking completion status. After they select a student, they can manage grades and absences by adding and editing them, and if they wish, they have the option to leave additional comments on each.
For parents, it shows the most recently added grades, absences, and homework. They can also view each category in more detail. If parents have multiple children in the school, they can view each of their records with the same account.

User Stories

- As a staff member, I want to log in to the system so that I can access and manage student information.
- As a staff member, I want to enter and update student grades so that I can keep accurate records of their academic performance.
- As a staff member, I want to record student absences so that attendance records are accurate and updated.
- As a staff member, I want to verify the excuses for student absences so that valid reasons for absences are documented.
- As a staff member, I want to manage the completion of homework so that parents are aware of their child's out-of-school tasks.
- As a parent, I want to log in to the system so that I can monitor my child's progress and school activities.
- As a parent, I want to view my child's grades so that I can stay informed about their academic progress.
- As a parent, I want to view my child's attendance records so that I can be aware of any absences.
- As a parent, I want to submit excuses for my child's absences so that the school is informed.

Use Case Diagram



Scenario Tables

Create Account

Use case name: Create Account	**UniqueID:** SMS001
Area: School management system	
Actor(s): Teacher, parent, account database	
Description: Allows teachers and parents to create an account for the system	
Triggering Event: The teacher/parent is on the login page and wants access to the system but does not own an account	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1. Teacher/parent selects the “Create account” option on the login page	Login, Account
2. Teacher/parent enters their email address in the ‘Email’ field	email, emailField
3. Teacher/parent enters the password they want to use in the ‘Password’ field	password, passwordField
4. Teacher/parent selects the account type	accType
5. Teacher/parent adds additional personal information	fullName, nameField, contactNumber, contactNumberField, address, addressField
6. Teacher/parent selects the “Create account” button	Account, AccountDB
7. A confirmation message is displayed, and the teacher/parent is returned to the login page	Login, AccountDB
Preconditions: Teacher needs access to the system to manage their class. The parent needs access to the system to keep track of their child’s progress at school	
Postconditions: The teacher/parent has created an account. The account is added to the accounts database	
Assumptions: Teacher/parent has an email they can use for the account	
Requirements met: Teacher/parent can create an account	
Outstanding issues: The teacher needs to verify his account (via code from school)	
Priority: High	
Risk: Medium	

Log In

Use case name: Log In	**UniqueID:** SMS002
Area: School management system	
Actor(s): Teacher, parent, account database	
Description: Allows teachers and parents to log into their accounts and use the system	
Triggering Event: The teacher/parent is at the login page and wants to access their account	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1. Teacher/parent is at the login page	Login
2. Teacher/parent enters their account email in the 'Email' field	email, emailField
3. Teacher/parent enters their password in the 'Password' field	password, passwordField
4. Teacher/parent clicks the 'Login' button	Login
5. System verifies that the email and password match an account	AccountDB, Login, email, hashedPassword
6. Teacher/parent gains access to the system	MainMenu
Preconditions: Teacher/parent must have an account	
Postconditions: Teacher/parent gets access to their account	
Assumptions: The account is correctly stored in the database, teacher/parent enters the correct account information	
Requirements met: Allows teacher/parent to log into the system using their account	
Outstanding issues: Number of unsuccessful attempts before additional action is required	
Priority: High	
Risk: Medium	

Manage Absence

Use case name: Manage Absence	UniqueID: SMS003
Area: School management system	
Actor(s): Teacher, absences database	
Description: Allows teacher to mark a specific student as being absent from school	
Triggering Event: The student does not attend class, the teacher needs to address it	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1. Teacher logs in	Login, email, password, emailField, passworldField
2. System verifies account information	AccountDB, Login, email, hashedPassword
3. Teacher select student from list	MainMenu, Student, StudentDB
4. Teacher clicks 'Add/Remove/Edit absences' button	Student, Absence
5. Teacher enters absence details in specified fields	absenceDate, a_comment
6. Teacher clicks 'Submit' button	MainMenu, Absence, AbsenceDB
7. Confirmation message is displayed	MainMenu
Preconditions: Student is absent from class	
Postconditions: Absence is recorded in the system	
Assumptions: Correct student is selected, correct date is entered, teacher is adding a new absence	
Requirements met: Allows teacher to keep records of student absences	
Outstanding issues: Absence database is not updated correctly	
Priority: Medium	
Risk: Low	

Manage Excuse

Use case name: Manage Excuse	**UniqueID:** SMS004
Area: School Management System	
Actor(s): Parent, excuse database	
Description: Allows parents to add excuses for absences	
Triggering Event: The student misses a class and the parent has to address it	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1.Parent logs into the system	Login, email, emailField, password, passwordField
2.System verifies account information	AccountDB, Login, email, hashedPassword
3. Parent selects child from list	MainMenu, Student, StudentDB
4.Parent clicks 'Absences' button	Absence, MainMenu
5. Parent selects absence from list	Absence, AbsenceDB
6.Parent enters excuse details in specified fields	Excuse, e_date, description, e_comment
7 . Parent clicks 'Submit' button	MainMenu, Excuse, ExcuseDB
8. Confirmation message is displayed	MainMenu
Preconditions: Parent needs to send an excuse for their child's absence	
Postconditions: Absence gets saved successfully in the database	
Assumptions: Parent is aware of child absence	
Requirements met: Parents can quickly add a reason for their child's absence	
Outstanding issues: Unclear guidelines for different types of excuses (medical, family emergency, etc.)	
Priority: Medium	
Risk: Low	

Manage Homework

Use case name: Manage Homework	**UniqueID:** SMS005
Area: School management system	
Actor(s): Teacher, homework database	
Description: Allows teachers to add/edit homework for the entire class	
Triggering Event: The teacher needs to update, create, or review homework assignments for their class.	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1. Teacher logs into the system	Login, email, emailField, password, passwordField
2. System verifies account info is valid	AccountDB, Login, email, hashedPassword
3. Teacher selects 'Add/Remove/Edit homework' button	Homework
4. Teacher adds homework details in specified fields	Homework, dueDate, title, description, subject
5. Teacher clicks 'Submit' button	MainMenu, Homework, HomeworkDB
6. Confirmation message is displayed	MainMenu
Preconditions: Teacher needs to add homework to his class	
Postconditions: Homework is successfully saved to the database	
Assumptions: Teacher is logged in	
Requirements met: Teachers can easily input, update, and manage homework assignments	
Outstanding issues: Data is incorrectly saved in database	
Priority: Medium	
Risk: Low	

View Homework

Use case name: View Homework	**UniqueID:** SMS006
Area: School management system	
Actor(s): Parent, homework database	
Description: Allows parents current homework assignments	
Triggering Event: Parent wants to check homework for their child	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1.Parent logs into the system	Login, email, emailField, password, passwordField
2. System verifies account information	AccountDB, Login, email, hashedPassword
3.Parent selects child from list	Student, StudentDB, MainMenu
4.Parent clicks the 'Homework' button	Homework, MainMenu
5.Parent is presented with homework assignments	Homework, HomeworkDB
6.Parent may select a homework to view additional information	Homework
Preconditions: Class has homework stored in database	
Postconditions: Homework is viewable on the system	
Assumptions: Homework is up to date in the system	
Requirements met: Parents can view homework assignments	
Outstanding issues: Missed homework visibility (parents might not always check the homework tab)	
Priority: Medium	
Risk: Low	

Manage Grade

Use case name: Manage Grade	**UniqueID:** SMS007
Area: School management system	
Actor(s): Teacher, grades database	
Description: Allows teachers to add/edit grades to students	
Triggering Event: Student received grade and teacher needs to address it	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1.Teacher logs into the system	Login, email, emailField, password, passwordField
2.System verifies account information	AccountDB, Login, email, hashedPassword
3.Teacher selects student from list	Student, StudentDB, MainMenu
4.Teacher clicks the 'Add grade' button	MainMenu, Grade
5.Teacher enters grade details	Grade, subject, mark, dateAdded, g_comment
6.Teacher clicks 'Submit button	Grade, GradeDB
7. Confirmation message is displayed	MainMenu
Preconditions: Teacher is logged in	
Postconditions: Grade is visible on student's profile, grades database is updated	
Assumptions: Correct grade details were entered	
Requirements met: Teacher can add/edit grades to students	
Outstanding issues: Handling invalid grade entries	
Priority: High	
Risk: High	

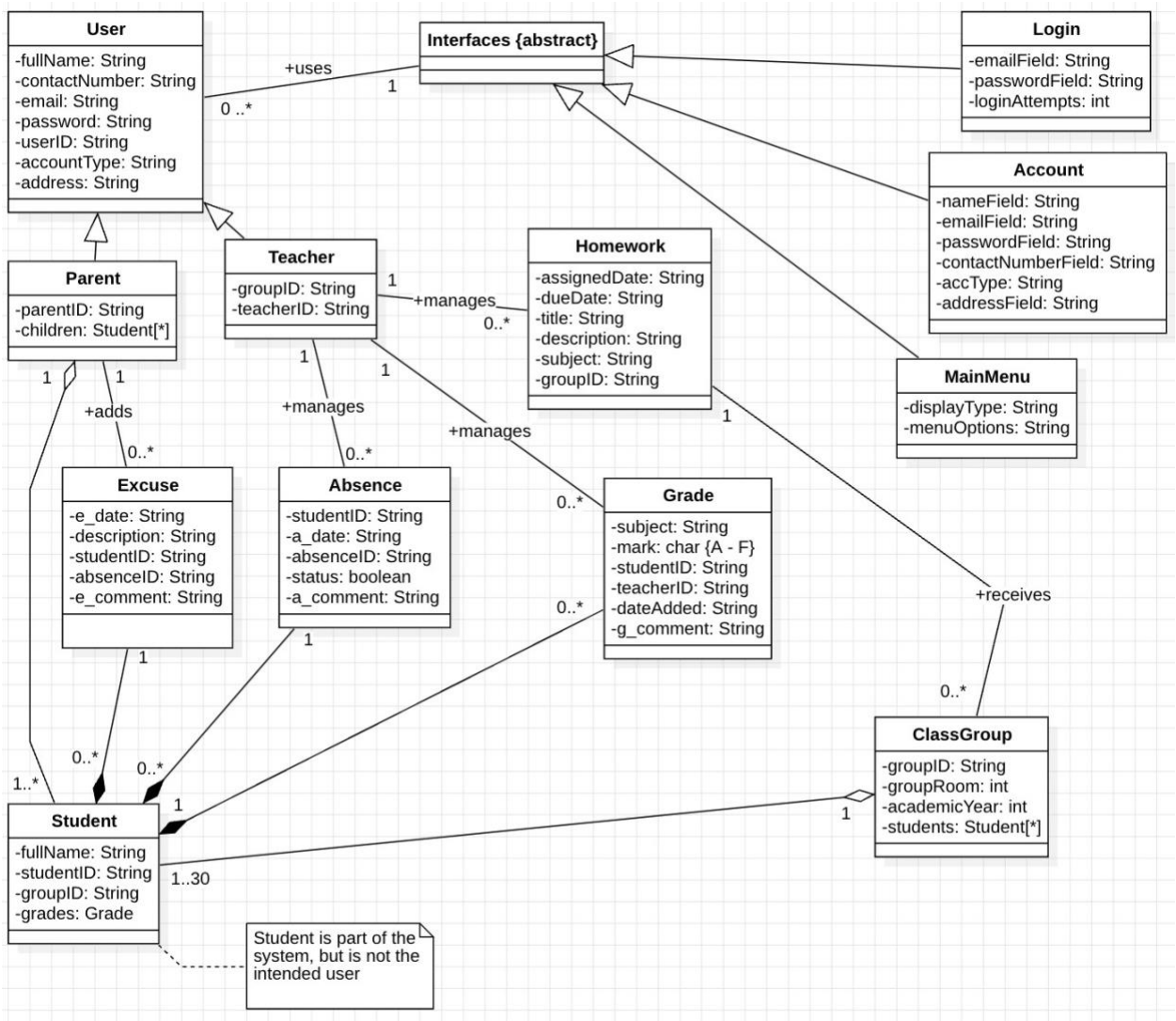
View Grade

Use case name: View Grade	**UniqueID:** SMS008
Area: School management system	
Actor(s): Parent, grades database	
Description: Allows parents to view grades of their children	
Triggering Event: Parents wants to check their child's current progress	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1.Parent logs into the system	Login, email, emailField, password, passwordField
2. System verifies account information	Login, AccountDB, email, hashedPassword
2.Parent selects child from list	Student, StudentDB, MainMenu
3.Parent clicks the 'View Grades button	MainMenu, Grade
4.Parent is presented with grades list	Grade, GradeDB
5.Parent may select specific grades to view details	Grade
Preconditions: Parent is logged in	
Postconditions: Grade is visible on child's profile	
Assumptions: Correct child was selected	
Requirements met: Parent can view their child's grades	
Outstanding issues: Filtering of grades	
Priority: High	
Risk: Medium	

Mark Absence

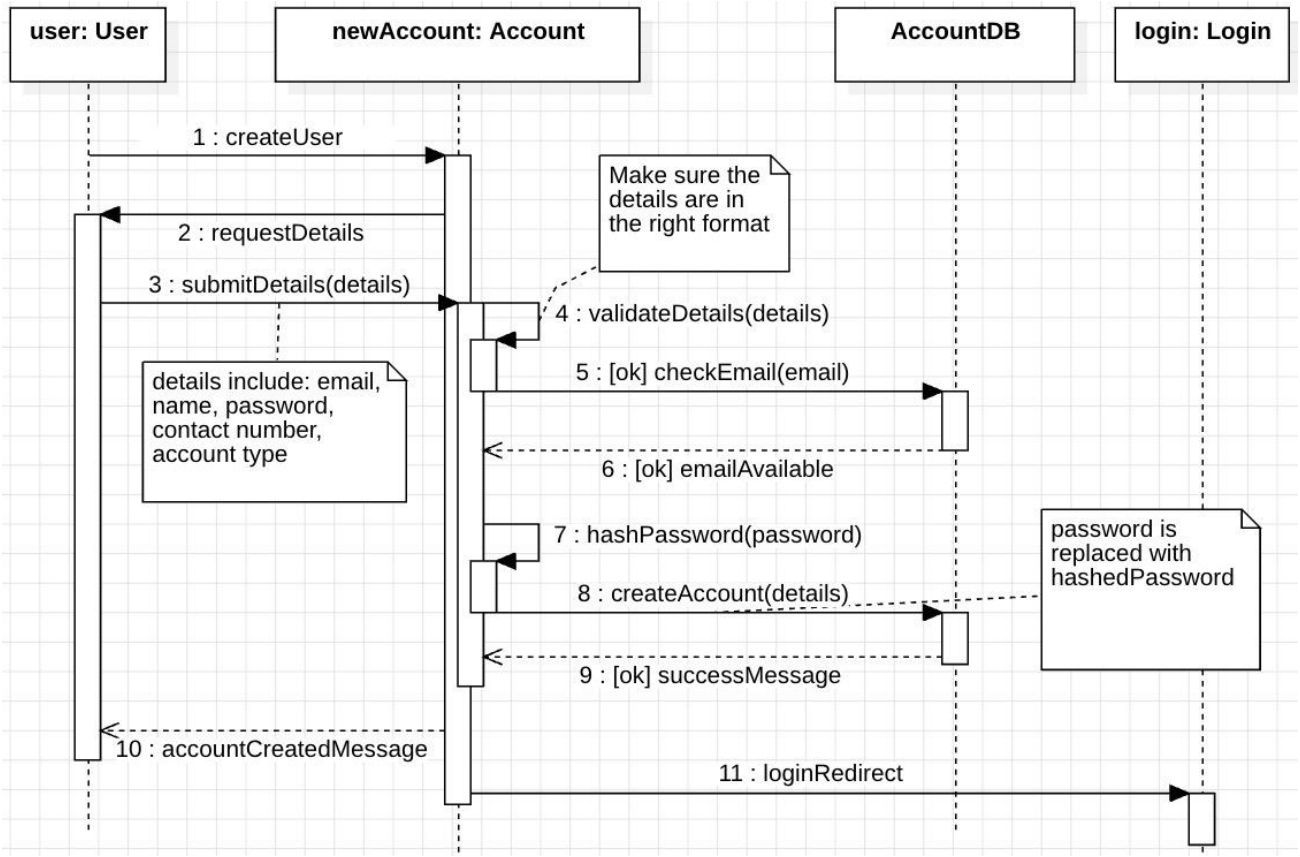
Use case name: Mark Absence	**UniqueID:** SMS009
Area: School management system	
Actor(s): Teacher, excuse database	
Description: Allows teachers to mark an excuse as valid or invalid for a student's absence.	
Triggering Event: The teacher needs to verify and mark an excuse provided by a parent.	
Trigger type: ☒**External** ☐ **Temporal**	
Steps Performed (Main Path)	**Information for Steps**
1. Teacher logs into the system	Login, email, emailField, password, passwordField
2. System verifies account info is valid	AccountDB, Login, email, hashedPassword
3. Teacher selects student from the list	MainMenu, Student, StudentDB
4. Teacher clicks the 'Absences' button	Absence, MainMenu
5. Teacher selects the absence with the excuse they want to mark	Absence, AbsenceDB
6. Teacher clicks the 'Mark Excuse' button.	Absence, status
7. Teacher selects either 'Valid' or 'Invalid' from the options.	Absence, status
8. Confirmation message is displayed	MainMenu, AbsenceDB
Preconditions: Teacher needs to mark an excuse	
Postconditions: The excuse is marked as valid or invalid in the system.	
Assumptions: Teacher is logged in	
Requirements met: Teachers can verify and mark excuses for student absences.	
Outstanding issues: Developing clear guidelines for what constitutes a valid or invalid excuse.	
Priority: Medium	
Risk: Low	

Class diagram



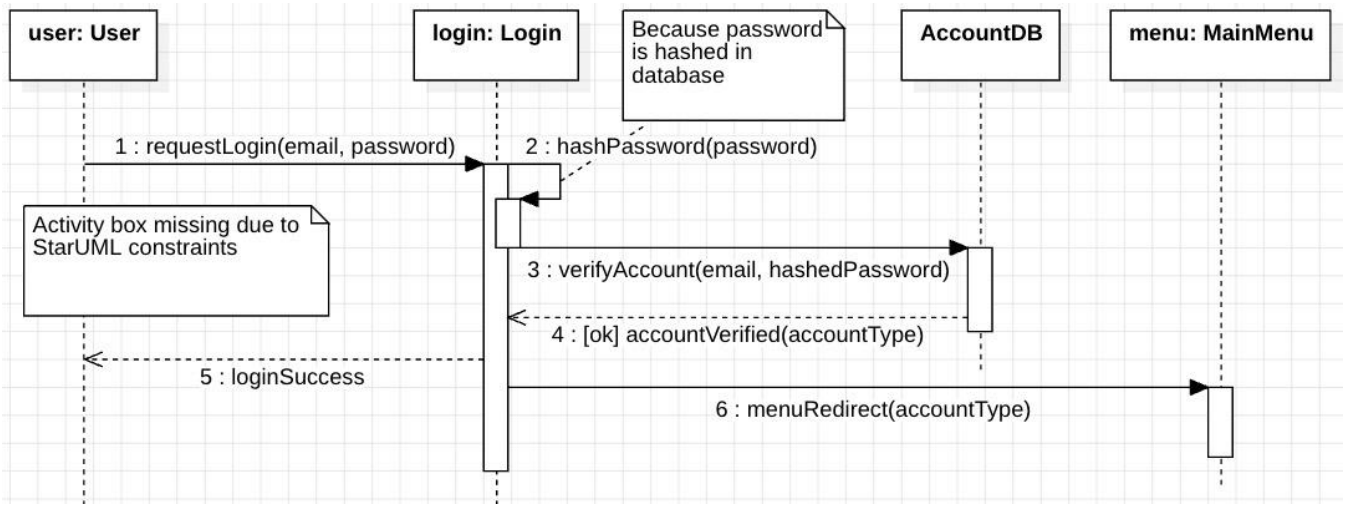
Sequence Diagrams

Create Account



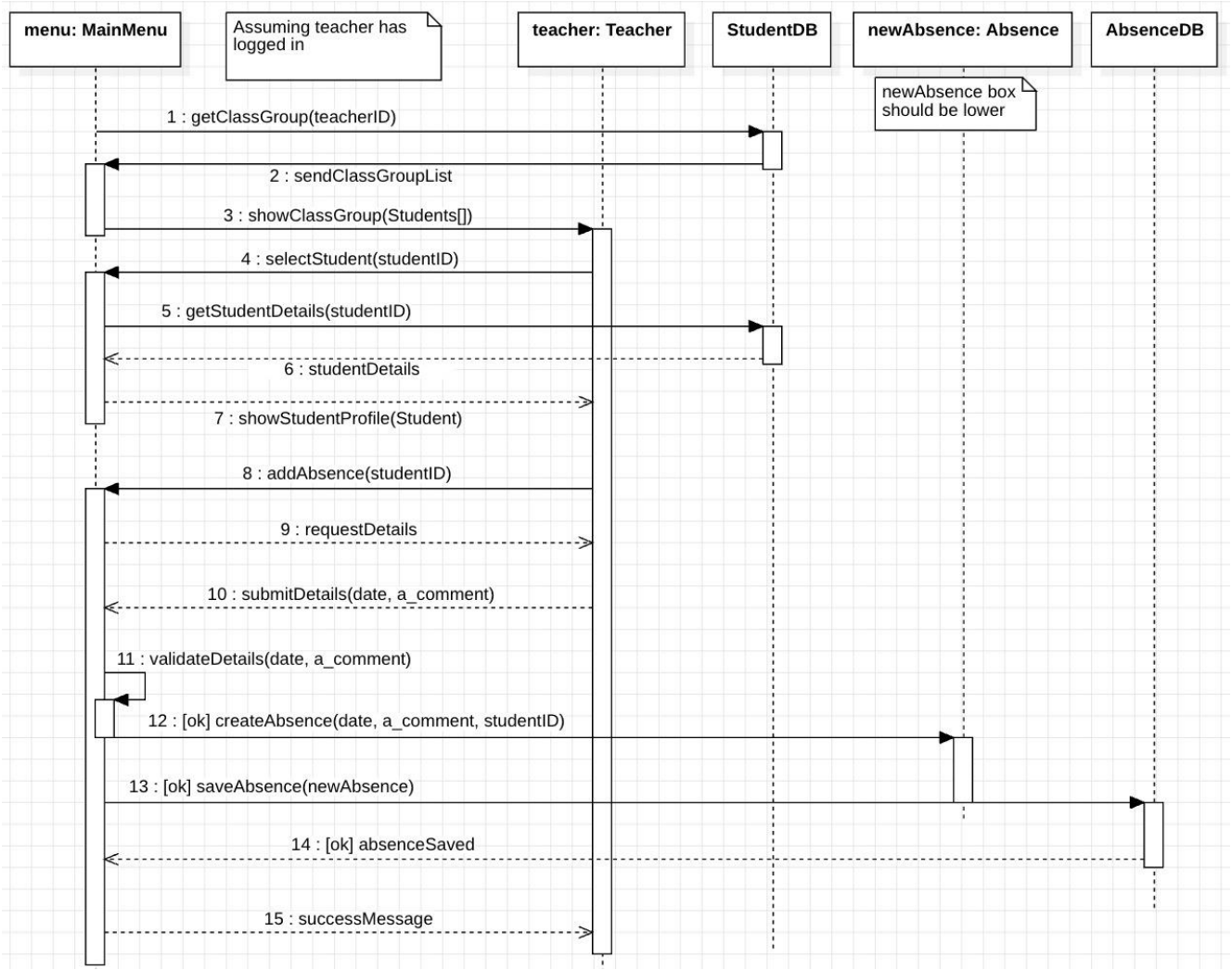
Sequence Number	Caller	Callee	Message Name	Message Type	Message Parameter	Message Constraints
1.	user	newAccount	createUser	Synchronous	-	-
2.	newAccount	user	requestDetails	Synchronous	-	-
3.	user	newAccount	submitDetails	Synchronous	email, password, fullName, contactNumber, accountType	-
4.	newAccount	newAccount	validateDetails	Synchronous	email, password, fullName, contactNumber, accountType	-
5.	newAccount	AccountDB	checkEmail	Synchronous	email	= if validation was successful
6.	AccountDB	newAccount	emailAvailable	Synchronous	-	= if email is available
7.	newAccount	newAccount	hashPassword	Synchronous	password	-
8.	newAccount	AccountDB	createAccount	Synchronous	email, hashedPassword, fullName, contactNumber, accountType	-
9.	AccountDB	newAccount	successMessage	Synchronous	-	= if account was added to database
10.	newAccount	user	accountCreatedMessage	Synchronous	-	-
11.	newAccount	login	loginRedirect	Synchronous	-	-

Log In



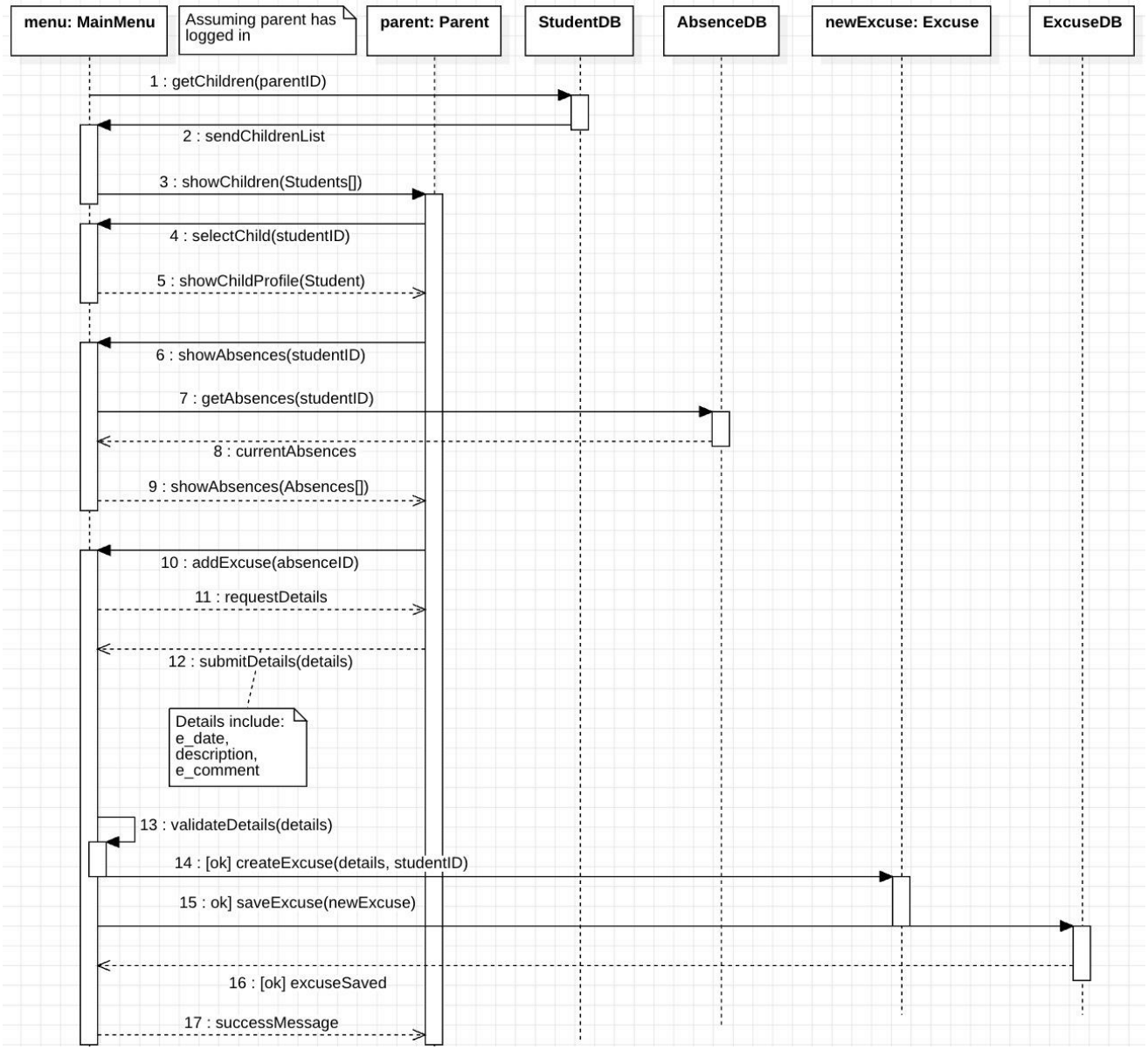
Sequence Number	Caller	Callee	Message Name	Message Type	Message Parameter	Message Constraints
1.	user	login	loginRequest	Synchronous	email, password	-
2.	login	login	hashPassword	Synchronous	password	-
3.	login	AccountDB	verify	Synchronous	email, hasedPassword	-
4.	AccountDB	login	accountVerified	Synchronous	accountType	= if account is in database
5.	login	user	loginSuccess	Synchronous	-	-
6.	login	menu	loginRedirect	Synchronous	-	-

Manage Absence



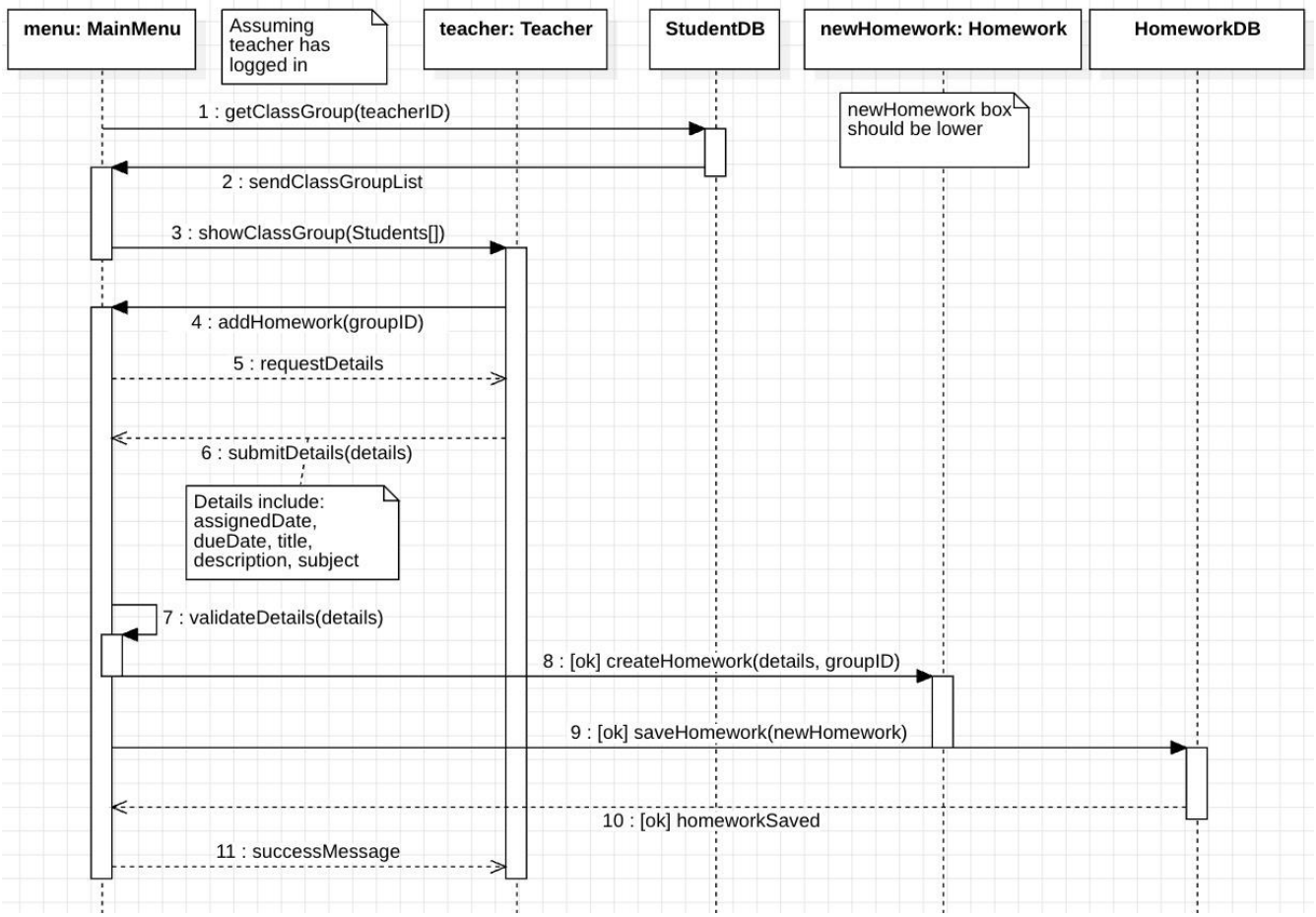
Sequence number	Caller	Callee	Message Name	Message Type	Message Parameter	Message Constraint
1.	menu	StudentDB	getClassGroup	Synchronous	teacherID	-
2.	StudentDB	menu	sendClassGroupList	Synchronous	-	-
3.	menu	teacher	showClassGroup	Synchronous	Students[]	-
4.	teacher	menu	selectStudent	Synchronous	studentID	-
5.	menu	StudentDB	getStudentDetails	Synchronous	studentID	-
6.	StudentDB	menu	studentDetails	Synchronous	-	-
7.	menu	teacher	showStudentProfile	Synchronous	Student	-
8.	teacher	menu	addAbsence	Synchronous	studentID	-
9.	menu	teacher	requestDetails	Synchronous	-	-
10.	teacher	menu	submitDetails	Synchronous	date, a_comment	-
11.	menu	menu	validateDetails	Synchronous	date, a_comment,	-
12.	menu	newAbsence	createAbsence	Synchronous	Date, a_comment, studentID	= if validation was successful
13.	menu	AbsenceDB	saveAbsence	Synchronous	newAbsence	= if newAbsence was create successfully
14.	AbsenceDB	menu	absenceSaved	Synchronous	-	= if absence was added to database
15.	menu	teacher	successMessage	Synchronous	-	-

Manage Excuse



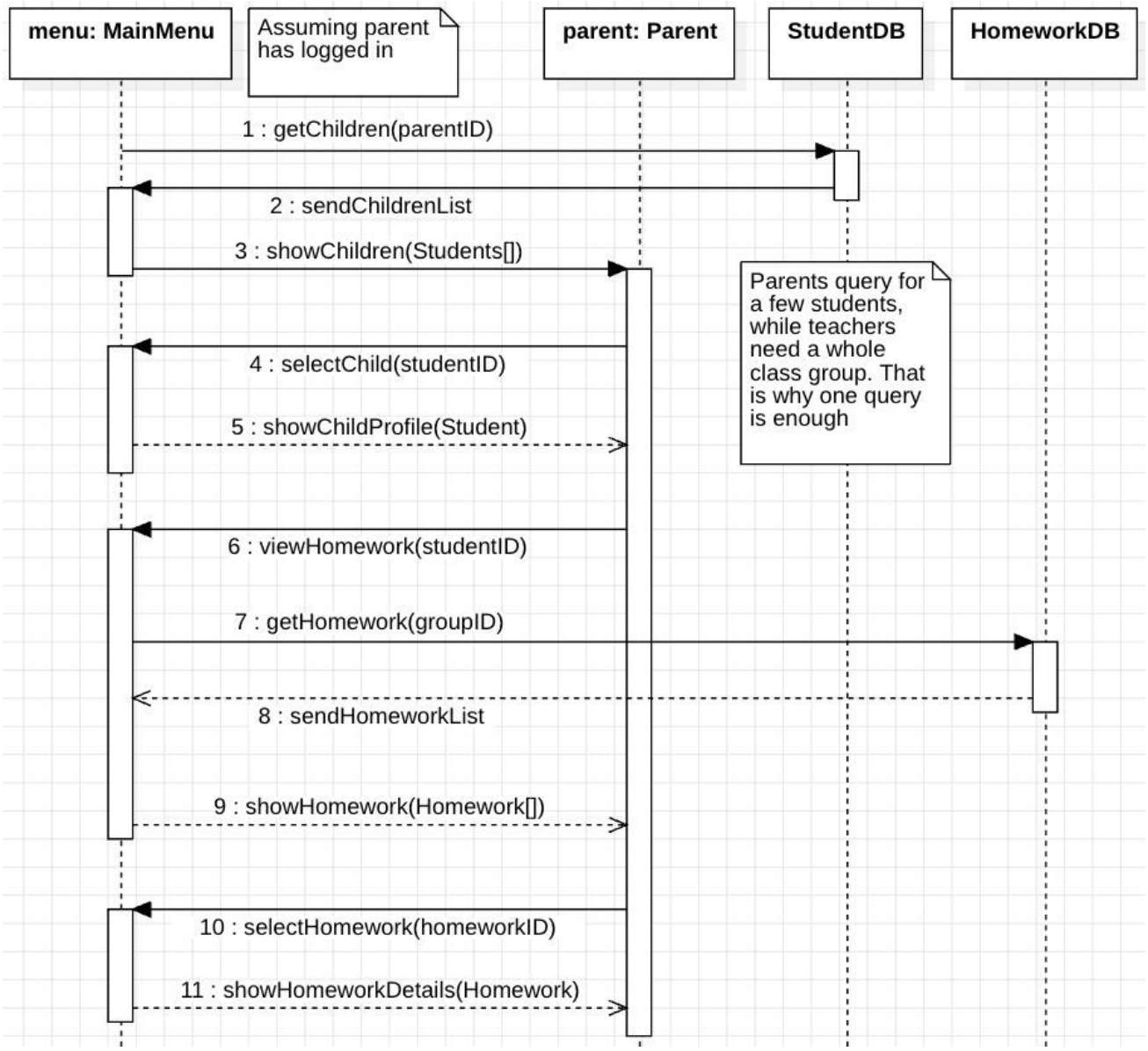
Sequence Number	Caller	Callee	Message Name	Message Type	Message Parameters	Message Constraints
1.	menu	StudentDB	getChildren	Synchronous	parentID	-
2.	StudentDB	menu	sendChildrenList	Synchronous	-	-
3.	menu	parent	showChildren	Synchronous	Students[]	-
4.	parent	menu	selectChild	Synchronous	studentID	-
5.	menu	parent	showChildProfile	Synchronous	Student	-
6.	parent	menu	showAbsences	Synchronous	studentID	-
7.	menu	AbsenceDB	getAbsences	Synchronous	studentID	-
8.	AbsencesDB	menu	currentAbsences	Synchronous	studentID	= absences with 'false' status
9.	menu	parent	showAbsences	Synchronous	Absences[]	-
10.	parent	menu	addExcuse	Synchronous	absenceID	-
11.	menu	parent	requestDetails	Synchronous	-	-
12.	parent	menu	submitDetails	Synchronous	e_date, description, e_comment	-
13.	menu	menu	validateDetails	Synchronous	e_date, description, e_comment	-
14.	menu	newExcuse	createExcuse	Synchronous	date, description, e_comment, studentID	= if validation was successful
15.	menu	ExcuseDB	saveExcuse	Synchronous	newExcuse	= if excuse was created successfully
16.	ExcuseDB	menu	excuseSaved	Synchronous	-	= if excuse was added to database
17.	menu	parent	successMessage	Synchronous	-	-

Manage Homework



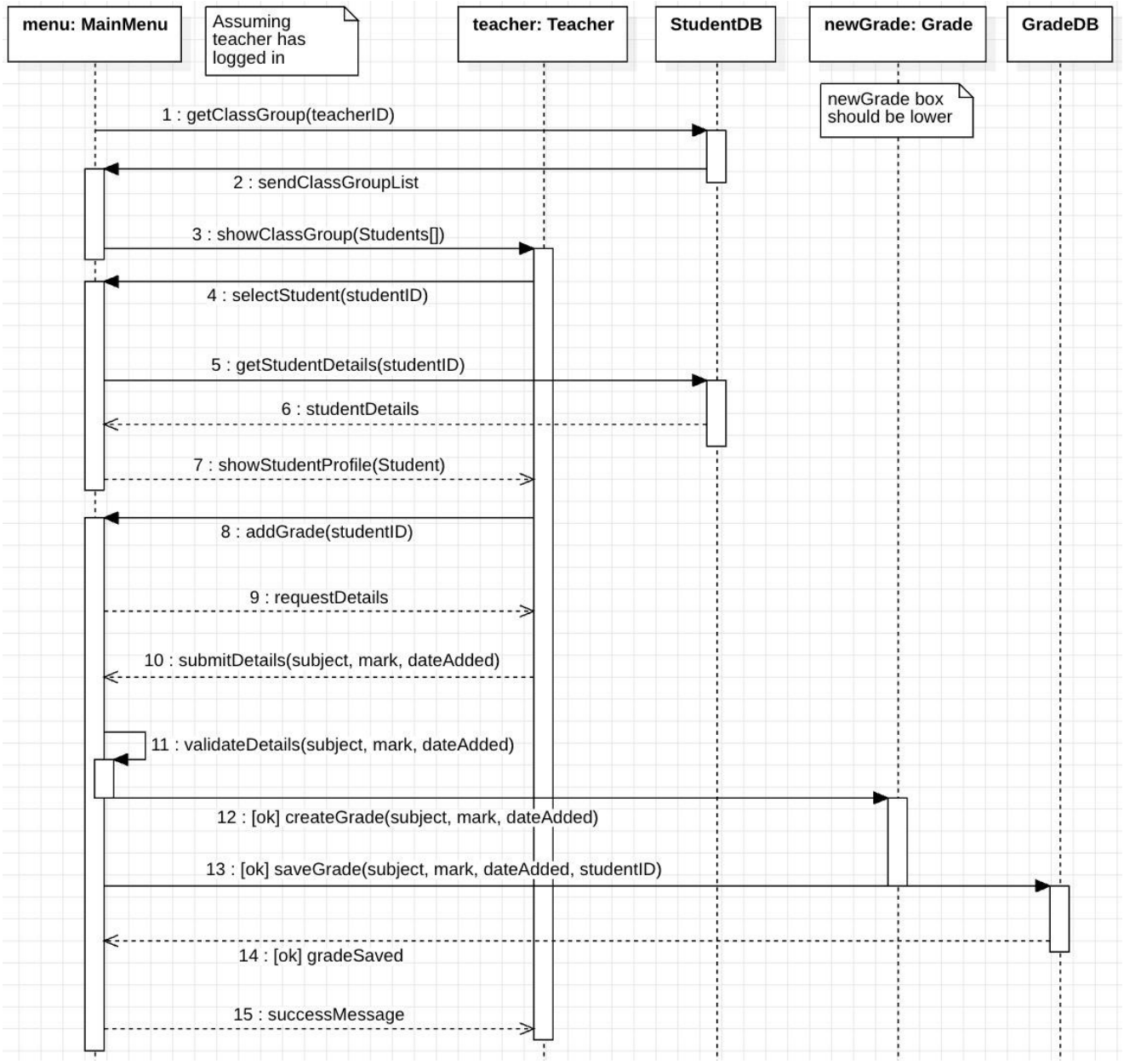
Sequence Number	Caller	Callee	Message Name	Message Type	Message Parameters	Message Constraints
1.	menu	StudentDB	getClassGroup	Synchronous	teacherID	-
2.	StudentDB	menu	sendClassGroupList	Synchronous	-	-
3.	menu	teacher	showClassGroup	Synchronous	Students[]	-
4.	teacher	menu	addHomework	Synchronous	groupID	-
5.	menu	teacher	requestDetails	Synchronous	-	-
6.	teacher	menu	submitDetails	Synchronous	assignedDate, dueDate, title, description, subject	-
7.	menu	menu	validateDetails	Synchronous	assignedDate, dueDate, title, description, subject	-
8.	menu	newHomework	createHomework	Synchronous	assignedDate, dueDate, title, description, subject, groupID	= if validation was successful
9.	menu	HomeworkDB	saveHomework	Synchronous	newHomework	= if homework was created successfully
10.	HomeworkDB	menu	homeworkSaved	Synchronous	-	= if homework was added to database
11.	menu	teacher	successMessage	Synchronous	-	-

View Homework



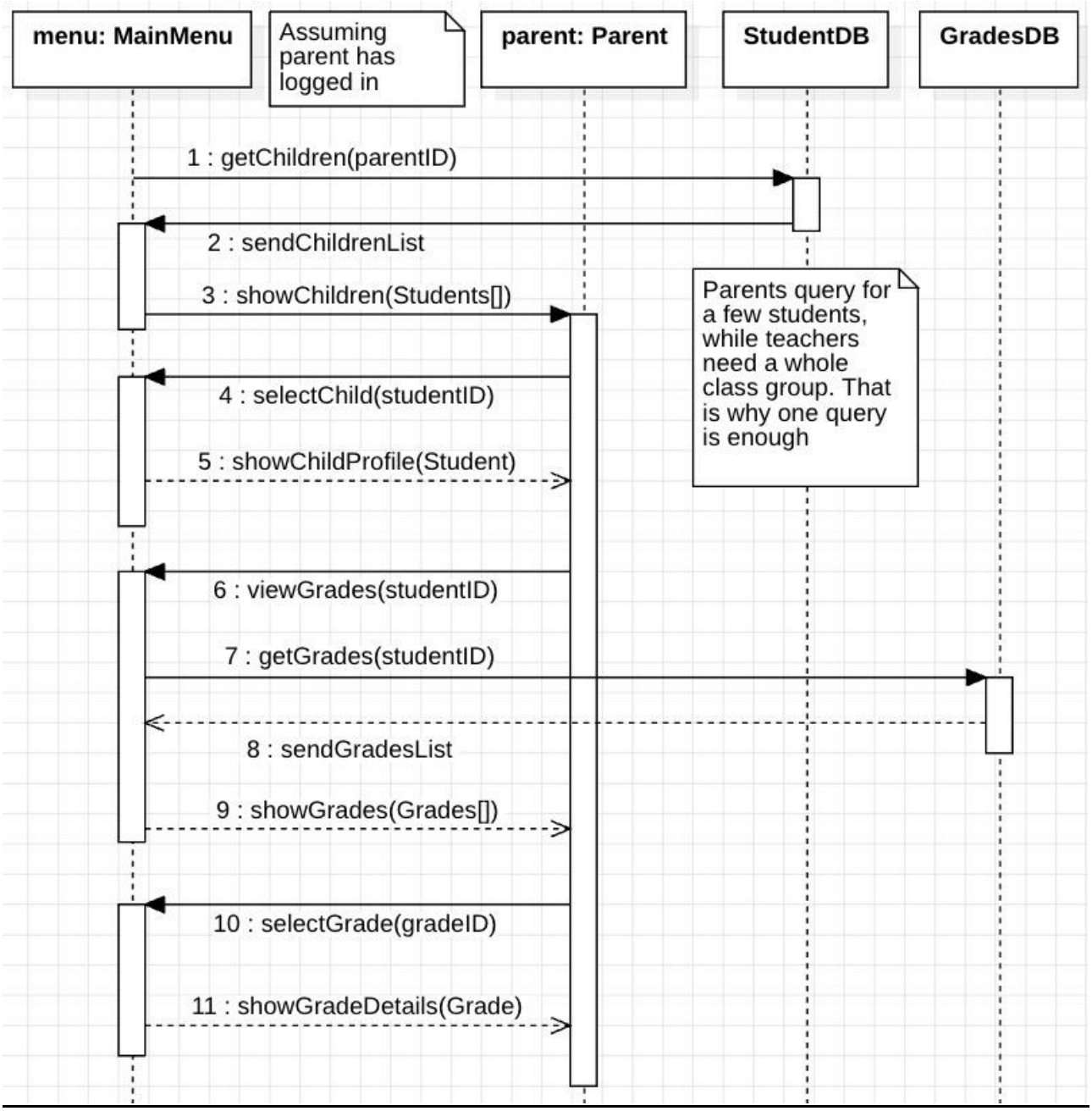
Sequence Number	Caller	Callee	Message Name	Message Type	Message Parameters	Message Constraints
1.	menu	StudentDB	getChildren	Synchronous	parentID	-
2.	StudentDB	menu	sendChildrenList	Synchronous	-	-
3.	menu	parent	showChildren	Synchronous	Students[]	-
4.	parent	menu	selectChild	Synchronous	studentID	-
5.	menu	parent	showChildProfile	Synchronous	Student	-
6.	parent	menu	viewHomework	Synchronous	studentID	-
7.	menu	HomeworkDB	getHomework	Synchronous	groupID	-
8.	HomeworkDB	menu	sendHomeworkList	Synchronous	-	-
9.	menu	parent	showHomework	Synchronous	Homework[]	-
10.	parent	menu	selectHomework	Synchronous	homeworkID	-
11.	menu	parent	showHomeworkDetails	Synchronous	Homework	-

Manage Grade



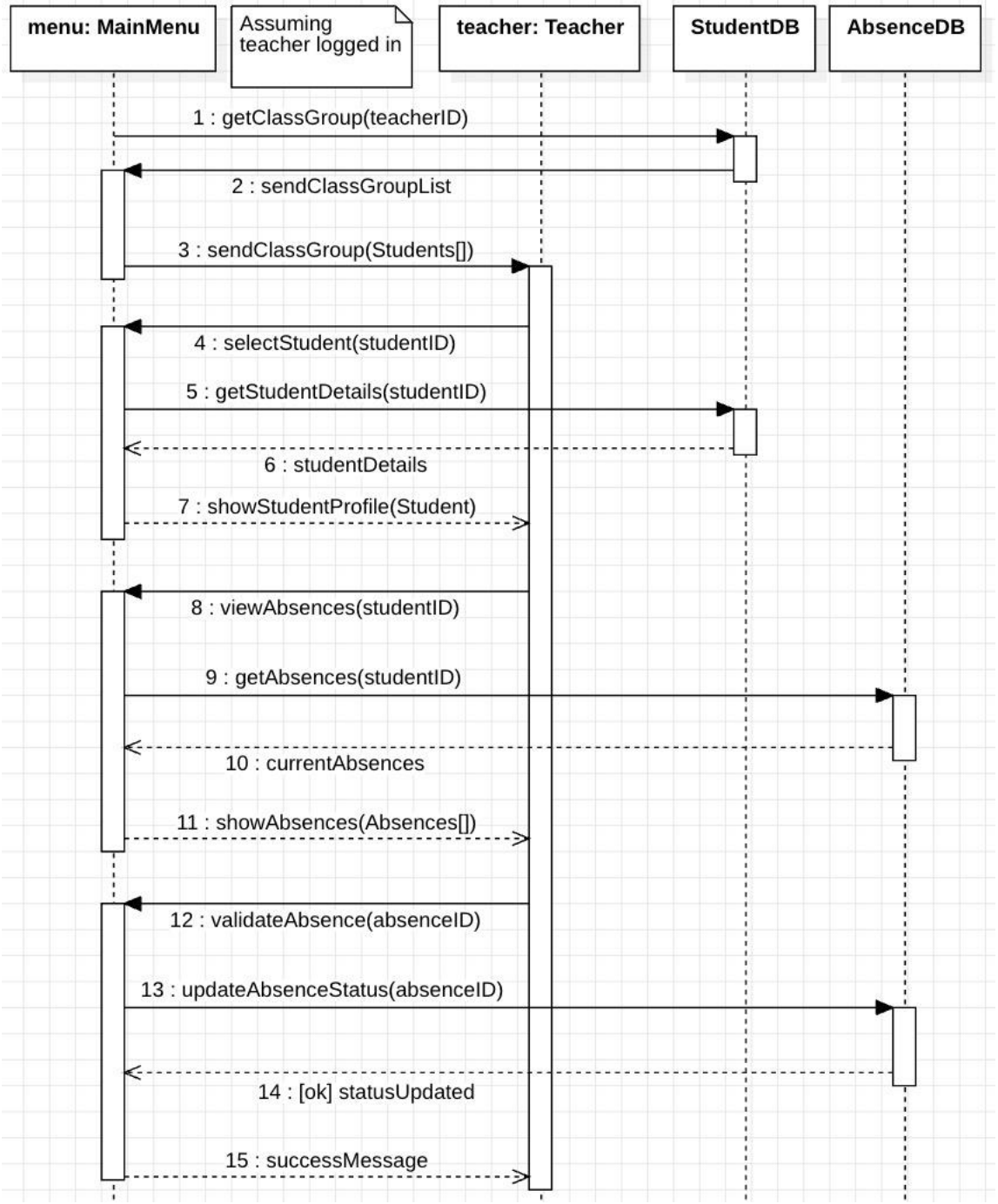
Sequence Number	Caller	Callee	Message Name	Message Type	Message Parameters	Message Constraints
1.	menu	StudentDB	getClassGroup	Synchronous	teacherID	-
2.	StudentDB	menu	sendClassGroupList	Synchronous	-	-
3.	menu	teacher	showClassGroup	Synchronous	Students[]	-
4.	teacher	menu	selectStudent	Synchronous	studentID	-
5.	menu	StudentDB	getStudentDetails	Synchronous	studentID	-
6.	StudentDB	menu	studentDetails	Synchronous	-	-
7.	menu	teacher	showStudentProfile	Synchronous	Student	-
8.	teacher	menu	addGrade	Synchronous	studentID	-
9.	menu	teacher	requestDetails	Synchronous	-	-
10.	teacher	menu	submitDetails	Synchronous	subject, mark, dateAdded	-
11.	menu	menu	validateDetails	Synchronous	subject, mark, dateAdded	-
12.	menu	newGrade	createGrade	Synchronous	subject, mark, dateAdded	= if validation was successful
13.	menu	GradeDB	saveGrade	Synchronous	newGrade	= if grade was created successfully
14.	GradeDB	menu	gradeSaved	Synchronous	-	= if grade was added to database
15.	menu	teacher	successMessage	Synchronous	-	-

View Grade


menu: MainMenu
Assuming parent has logged in
parent: Parent
StudentDB
GradesDB
1 : getChildren(parentID)
2 : sendChildrenList
3 : showChildren(Students[])
Parents query for a few students, while teachers need a whole class group. That is why one query is enough
4 : selectChild(studentID)
5 : showChildProfile(Student)
6 : viewGrades(studentID)
7 : getGrades(studentID)
8 : sendGradesList
9 : showGrades(Grades[])
10 : selectGrade(gradeID)
11 : showGradeDetails(Grade)

Sequence Number	Caller	Callee	Message Name	Message Type	Message Parameters	Message Constraints
1.	menu	StudentDB	getChildren	Synchronous	parentID	-
2.	StudentDB	menu	sendChildrenList	Synchronous	-	-
3.	menu	parent	showChildren	Synchronous	Students[]	-
4.	parent	menu	selectChild	Synchronous	studentID	-
5.	menu	parent	showChildProfile	Synchronous	Student	-
6.	parent	menu	viewGrades	Synchronous	studentID	-
7.	menu	GradesDB	getGrades	Synchronous	studentID	-
8.	GradesDB	menu	sendGradesList	Synchronous	-	-
9.	menu	parent	showGrades	Synchronous	Grades[]	-
10.	parent	menu	selectGrade	Synchronous	gradeID	-
11.	menu	parent	showGradeDetails	Synchronous	Grade	-

Mark Absence



Sequence Number	Caller	Callee	Message Name	Message Type	Message Parameters	Message Constraints
1.	menu	StudentDB	getClassGroup	Synchronous	teacherID	-
2.	StudentDB	menu	sendClassGroupList			-
3.	menu	teacher	sendClassGroup		Students[]	-
4.	teacher	menu	selectStudent		studentID	-
5.	menu	StudentDB	getStudentDetails		studentID	-
6.	StudentDB	menu	studentDetails		-	-
7.	menu	teacher	showStudentProfile		Student	-
8.	teacher	menu	viewAbsences		studentID	-
9.	menu	AbsenceDB	getAbsences		studentID	-
10.	AbsenceDB	menu	curretnAbsences		-	-
11.	menu	teacher	showAbsences		Absences[]	-
12.	teacher	menu	validateAbsence		absenceID	-
13.	menu	absenceDB	updateAbsenceStatus		absenceID	-
14.	AbsenceDB	menu	statusUpdated		-	= if database was updated successfully
15.	menu	teacher	successMessage		-	-

Test Cases

Create Account

Name	CC-21(The User has to create an account)
Requirement	MR-41(email requirements met e.g. having a @)
Preconditions	The user needs an account and has a email and password ready
Steps	1.Press the “Create New Account” button on the Main Menu screen 2.Select is it a Student/Parent or Teacher account 3.Enter the email “BobCoushotmail.com” when prompted 4.Enter a password “Ibon234!” 5.Add the extra personal info when the fields are on the screen (e.g. Emergency Num 089 213 4567,Address 12 Ash Grove etc) 6.Press “Finish Account” to make the new account
Expected results	1.The Email should not be taken by the system missing “@” 2.Once the correct email is entered the account should be created a message “Account creation is Successful” should be displayed 3.The user should be logged in 3.Press log out and return to the main menu screen MM-25

Log In

Name	LG-25(The User has to Log In)
Requirement	MR-42(email requirements met e.g. having a @)
Preconditions	The user needs to have a account already created
Steps	1.Press the “Log In” button on the Main Menu screen 3.Enter the email “BobCous@hotmail.com” when prompted 4.Enter a password “Ibon234!” 6.Press the “Log In” button to Log In
Expected results	1.The Email and password should work 2.The user should be displayed a message telling them they logged in successfully 3.The user should be logged in 3.Press log out and return to the main menu screen MM-25

Manage Absence

Name	AA-22 Verify that the absence can be added to a student
Requirement	MR-22 (only numbers in a date format accepted for absence date and string for absence description)
Preconditions	The Teacher is logged into their account
Steps	1.Click on the “Student: Student Name” button from the list of students 2. Click on the “Add new Absence” button 3. Enter the absence date 12/23/23 4. Click on the “Description” field 5. Enter the absence description Student was not present during the first two classes 6. Click on the button “Add Absence”
Expected results	1. The Absence is accepted by the system 2. Verify that the absence is present on the student’s account 3. Return to class screen CS-12

Manage Excuse

Name	AE-31 (The Parent can add a excuse for a absence)
Requirement	MR-40 (only allowed to add excuse & not change the original absence)
Preconditions	The Parent is logged into their account
Steps	1.Click on the “Student: Student Name” button 2. Click on the “Absences” button 3. Click on the Absence that is to be excused (select the newest for the testing) 4. Click on the “Add Absence” button 5. Type in the absence into the input box 6. Click on “Submit Absence” to add it to the absence 7. Click on the “Return” button
Expected results	1. The Excuse is saved to the absence on the system 2. Verify that the Excuse can be viewed by the teacher as well 3. Return to the Student account screen SC-17

Manage Homework

Name	HW-22 Verify that the homework can be added to the class
Requirement	MR-26 (only the selected class should have the homework)
Preconditions	The Teacher is logged into their account
Steps	1.Click on the “Class: Class Name” button from the list of classes 2. Click on the “Add Homework” button 3. Enter the subject for the Homework 4. Click on the “Work” field 5. Enter the questions or any other work that is required for the homework 6. Click on the button “Post Homework”
Expected results	1. The Homework is accepted by the system 2. Verify that the Homework is present on every student’s account from the selected class 3. Return to class screen CS-12

View Homework

Name	VH-34 Verify that the Homework can be viewed by Parents and students
Requirement	MR-40 (only allowed to view homework and how much is completed)
Preconditions	The Parent is logged into their account
Steps	1.Click on the “Student: Student Name” button 2. Click on the “Homework” button 3. Click on the Homework that is to be checked from the list (select the newest for the testing) 4. Click on the “Work” button 5. The questions or any other work that is needed is presented alongside the work already done and posted for that homework assignment 6. Click on the “Return” button
Expected results	1. The Homework is present and accessible on the system 2. Verify that the work can be viewed by a Parent/Student account 3. Return to the Student account screen SC-17

Manage Grade

Name	AG-21 Verify that the grade can be added to a student
Requirement	MR-21 (only one capital letter for grade)
Preconditions	The Teacher is logged into their account
Steps	1.Click on the “Student: Student Name” button from the list of students 2. Click on the “Add new Grade” button 3. Enter the grade A 4. Click on the “Description” field 5. Enter the grade description This grade is for English 6. Click on the button “Submit Grade”
Expected results	1. The grade is accepted by the system 2. Verify that the grade is present on the student’s account 3. Return to class screen CS-12

View Grade

Name	VG-31 Verify that the grade can be viewed by Parent
Requirement	MR-31 (only allowed to view not edit)
Preconditions	The Parent is logged into their account
Steps	1.Click on the “Student: Student Name” button 2. Click on the “View Grades” button 3. Click on the newest grade from the list 4. Click on the “Description” button 5. The grade description This grade is from the English exam total:86/100 should be visible but not editable 6. Click on the button “Return”
Expected results	1. The Grade is present on the system 2. Verify that the Grade is not editable by a Parent account 3. Return to View Grades screen SG-13

Mark Absence

Name	AS – 27 Verify that the Teacher can mark an excuse as valid or invalid)
Requirement	MR-71 (Teacher can validate or invalidate excuses for student absences)
Preconditions	The parent has submitted an excuse for their child's absence, and the teacher is logged into their account.
Steps	1. Click on the “Student: [Student Name]” button. 2. Click on the ‘Absences’ button. 3. Click on the specific absence 4. Click on the ‘Mark Excuse’ button. 5. Select the appropriate option (‘Valid’ or ‘Invalid’). 6. Verify that a confirmation message is displayed, and the excuse status is updated in the system.
Expected results	1. The system accepts the teacher's login credentials and displays the main menu. 3. The teacher successfully navigates to the student’s profile and views the list of absences. 4. The teacher can select an absence and view its details. 5. The ‘Mark Excuse’ button displays options to mark the excuse as ‘Valid’ or ‘Invalid’. 6. The selected status (valid or invalid) is applied to the excuse, and a confirmation message is displayed.

Potential Improvements

While the current system covers core functionality, we discussed additional features to further enhance user experience. Due to limitations in our current knowledge and skill, we were unable to design and incorporate these features in the current project, but they remain potential areas for improvement. Some of the features discussed include:

Parent - Teacher Communication Portal:
Develop a messaging system within the platform, allowing parents and teachers to communicate directly regarding student progress, concerns, and upcoming events.

Homework Submission and Feedback:
Add a feature for students to submit homework online and receive feedback from teachers, streamlining the homework process and promoting timely feedback.

Login Cooldown:
A feature where after 5 failed attempts in a short period of time, the system would put the user under a 'cooldown period' and notify them of an email that was sent to help with the login process.

Password Resetting
Add an automated email service which would send emails to users notifying them of failed attempts at login with their account. The email would let the user know at which time the logins were attempted and would provide a link to securely reset the current account password, if necessary.

Contributions

Summary	Group effort
User stories	Thomas Radulescu
Use case diagram	Group effort (brainstorming), Kristain Kesar (diagram)
Scenarios	Kristian Kesar: Create account, Log in Thomas Radulsescu: View homework, Manage homework Filip Raguz: the rest
Class diagram	Group effort (brainstorming), Filip Raguz (diagram)
Sequence diagram	Group effort (brainstorming) Thomas Radulescu & Filip Raguz (diagrams)
Test cases	Kristian Kesar