

עבודת גמר
בתכנון ותכנות מערכות
במסלול "שירותי רשת" 2017

שם התלמיד: אביב בואנו

ת.ז: 314711045

שם המורה: מאיר דהאן

בית הספר: שש שנתי נחשון מ.א. חבל מודיעין

תודות

ראשית ברצוני להודות לחבר, רום גביש, על התמיכה והעזרה בבניית ממשק יפה ונוח לשימוש על פי אדיאולוגיות עיצוביות של גוגל - 'מטריאל'.

כמו כן, אני מודה לאמי אשר עזרה לי באפיון המערכת ועזרה בפתירת בעיות אלגוריתמיות.

תודה אחרונה וחשובה, למשפחתי המדהימה אשר סיפקה עבורי סביבת למידה תומכת מועדדת ושקטה.

תוכן עניינים

2	תודות.....
3	תוכן עניינים
4	מבוא.....
4	תאור הפרויקט.....
5	מטרות הפרויקט
5	קהל יעד
6	מבנה בסיס הנתונים.....
6	טבלאות מתוך מסד הנתונים
6	רשימת טבלאות:
6	קשרי גומלין מתוך מסד הנתוני (DSD):.....
16	מבנה התיקיות:
17	המחלקות:
17	תרשימי UML:
18	פירוט חתימות ותוכן המחלקות:
57	שירותי רשת:
59	מדריכים.....
65	ReadMe.....

תאור הפרויקט

מערכת לניהול בתי ספר אינטרנטית, המאפשרת מעבר מהיר של מידע בין אנשי הצוות בבתי ספר והמורים. המערכת נותנת לתלמיד דוחות על פעילותו הלימודית. המערכת מאפשרת למורה להזין ציונים והערות משמעת. בנוסף לכך יכול המורה לקבל דוחות על תלמיד באופן פרטני על פעילותו בשיעורים ובמבחנים על מנת למנף את הישגיו הלימודיים. המורה יכול גם לקבל דוח כיתתי על מנת לראות את מצב הכיתה במאקרו. המורים והתלמידים מקבלים מערכת אישית היכולה להציג שינויי מערכת פרטניים אליהם, לדוגמא, במידה והמורה דני הודיע להנהלה כי אינו יכול להגיע ביום מסויים יופיע לכל תלמידיו ביטול שיעור במערכת. סגל ההנהלה של בית הספר יכול להוסיף אנשי סגל ותלמידים, לייצור כיתות, להוסיף ו/או למחוק שיעורים, לשנות שיעורים לתאריך מסוים (ביטול, מילוי מקום ועוד), בנוסף סגל ההנהלה יכול לערוך את כרטיס התאמות של התלמיד.

מטרות הפרויקט

- הפקת דוחות נתוני לימוד לתלמידים.
- הפקת דוחות נתוני לימוד של התלמידים למורים.
- הפקת דוחות כניסה למערכת לסגל ההנהלה.
- ניהול מערכת שעות יותר נוח ונגיש לבתי ספר.
- ניהול התאמות התלמידים.
- נגישות המורה לתלמידים, למורה קיימת אפשרות חיפוש תלמידים אשר הוא מלמד.
- נגישות סגל ההנהלה לכל הסגל והתלמידים, לסגל ההנהלה קיימת אפשרות חיפוש אנשי סגל ותלמידים.

קהל היעד

מנהל – סגל ההנהלה הבית ספרי, מזכירות ומנהלים/מנהלות בית ספר.

מורה – כל מורה בבית הספר.

תלמיד – כל תלמיד בבית הספר.

מבנה מסד הנתונים

רשימת טבלאות

Tables

eduCities
eduDisciplines
eduDisciplinesMembers
eduExams
eduGrades
eduKnownSchools
eduLearnGroups
eduLessonChanges
eduLessons
eduMajors
eduMajorsMembers
eduMajorsTgrades
eduMembers
eduMessages
eduParentMember
eduSchools
eduScores
eduTeacherGrades

f

טבלת ערים (eduCities)

eduCityID	AutoNumber	מזהה עיר - City ID
eduCity	Short Text	שם עיר - City Name

eduCityID	eduCity
4	בוסריחאן (שבט)
15	אביאל
21	אבירים
32	אדמית
45	אור יהודה

טבלת סוגי הערות משמעת (eduDisciplines)

eduDisciplinesID	AutoNumber	מספר סוג הערת המשמעת
eduDisciplinesTitle	Short Text	שם סוג הערת המשמעת
eduDisciplinesScore	Number	ציון סוג הערת המשמעת
eduDisciplinesPositive	Yes/No	סוג הערת המשמעת חיובי/שלילי

+	eduDisciplinesTitle	eduDisciplinesScore	eduDisciplinesPositive
+	1 אי הכנת שיעורי בית	-5	<input type="checkbox"/>
+	2 אי הגשת עבודה	-10	<input type="checkbox"/>
+	3 אי הבאת ציוד	-5	<input type="checkbox"/>
+	4 חיסור	-3	<input type="checkbox"/>
+	5 איחור	-2	<input type="checkbox"/>

טבלת סוגי הערות משמעת – משתמשים (eduDisciplinesMembers)

eduLessonID	Number	מזהה השיעור
eduDisciplinesID	Number	מזהה סוג הערת המשמעת
eduStudentID	Number	מזהה התלמיד
eduDate	Date/Time	תאריך התרחשות

eduLessonID	eduDisciplinesID	eduStudentID	eduDate
3	1	176	25/12/2016
3	2	176	25/12/2016
3	2	176	1/1/2017
3	3	180	25/12/2016
3	3	180	1/1/2017

טבלת מבחנים (eduExams)

eduExamID	AutoNumber	מזהה מבחן
eduExamTitle	Short Text	כותרת מבחן
eduTeacherID	Number	מזהה מורה
eduExamDate	Date/Time	תאריך מבחן
eduTgradeID	Number	מזהה כיתה מקצוע
eduPrecent	Number	אחוז הציון של המבחן
eduYearPart	Short Text	סמסטר מבחן a/b

eduExamID	eduExamTitle	eduTeacherID	eduExamDate	eduTgradeID	eduPrecent	eduYearPart
12	מבחן מתיחה	159	13/10/2016	10	70	a
13	מבחן רקורסיה	159	10/1/2017	13	20	b
15	בוחן זמנים	159	1/1/2017	11	87	b
22	מבחן מבני נתונים	159	11/1/2017	13	70	b
26	מבחן רקורסיה	159	12/10/2016	14	40	a
27	מבחן רקורסיה	159	4/1/2017	14	50	b

טבלת כיתות אם (eduGrades)

eduGradeID	AutoNumber	מזהה כיתה אם
eduGradeName	Short Text	שם כיתה אם

eduGradeID	eduGradeName
1	י'1
2	י'2
3	י'3
4	יא'1
5	יא'2
6	מורה
7	מנהל
8	ז'1
9	ז'2
10	ז'3
11	יא'3
12	יא'4

טבלת בתי ספר מוכרים (eduKnownSchools)

eduSchoolID	Number	The id of the school
eduSchoolName	Short Text	The name of the school
eduSchoolCity	Number	The city of the school
eduSchoolManager	Short Text	The name of the school manager
eduMidSchoolManager	Short Text	The name of its mid school manager
eduSchoolLanguage	Short Text	The school teaching language
eduSchoolStudentCount	Number	The aprox amout of students

eduSchoolID	eduSchoolName	eduSchoolCity	eduSchoolManager	eduMidSchoolManager	eduSchoolLanguage	eduSchoolStudentCount
110023	בית יעקב מטרסדורף	2082	צפורה הייזלר		עברית	1024
110098	השחר	2029	שירלי סלמון		עברית	455
110106	אגרון גרשון	2082	דינה יריב		עברית	305
110130	סאלד	2082	מיעד יעקב מלצר		עברית	302
110155	בית הכרם	2082	מרב פאול		עברית	527
110171	גאולים א	2082	יאנה גרסון		עברית	325
110205	מ"מ רחביה ע"ש דוד ג	2082	ענת לביא		עברית	393

טבלת כיתות מקצוע – תלמיד (eduLearnGroups)

eduTgradeID	Number	מזהה כיתת מקצוע
eduStudentID	Number	מזהה משתמש - תלמיד
eduDate	Date/Time	תאריך חיבור תלמיד-כיתה

eduTgradeID	eduStudentID	eduDate
10	157	1/9/2016
10	176	1/9/2016
10	177	1/9/2016
10	186	1/9/2016
11	156	1/9/2016
11	180	1/9/2016
11	183	1/9/2016
11	187	1/9/2016
12	180	1/9/2016
12	187	1/9/2016

טבלת שינויים בשיעורים (eduLessonChanges)

eduLessonChangeID	AutoNumber	מזהה שינויי שיעור
eduLessonChangeType	Short Text	סוג שינויי
eduLessonID	Number	מזהה שיעור
eduMessage	Short Text	הודעת שינויי
eduDate	Date/Time	תאריך

eduLessonChangeID	eduLessonChangeType	eduLessonID	eduMessage	eduDate
3 c		16		9/4/2017
4 c		35		9/4/2017
5 c		3		9/4/2017
6 c		4		9/4/2017
7 c		11		9/4/2017
10 c		33		9/4/2017

טבלת שיעורים (eduLessons)

eduLessonID	AutoNumber	The id of the lesson - מזהה שיעור
eduTgradeID	Number	The teacher grade id for the lesson - מזהה כיתה מקצוע
eduDay	Number	יום - Day 1-7
eduHour	Number	שעה - Hour 1-12
eduActive	Short Text	מצב שיעור - Lesson State

eduLessonID	eduTgradeID	eduDay	eduHour	eduActive
3	10	1	1	Yes
4	10	1	2	Yes
19	10	3	5	No
20	10	2	3	Yes
32	10	1	3	Yes
33	10	1	4	Yes
34	10	1	5	Yes
5	11	3	2	Yes
8	11	3	3	Yes
9	11	3	4	Yes
11	11	1	3	Yes
10	12	3	1	Yes

טבלת מגמות (eduMajors)

eduMajorID	AutoNumber	מזהה מגמה
eduMajor	Short Text	שם מגמה

eduMajorID	eduMajor
1	מדעי המחשב
2	שירותי רשת
3	ביולוגיה
4	קולנוע
5	סוציולוגיה
6	כימיה
7	פיזיקה
8	ניהול עסקי
9	אומנות העיצוב
10	ערבית
11	ספורט
13	ליבה
19	עיצוב גנרי

טבלת מגמות - משתמשים (eduMajorsMembers)

eduUserID	Number	מספר המשתמש במערכת (מספור אוטומטי) - (A_I)
eduMajorID	Number	מספר המגמה במערכת (מספור אוטומטי) - (A_I)

eduUserID	eduMajorID
1	1
1	2
1	7
149	1
149	2
149	3
149	8
154	1
154	7
155	1
155	2
155	3
156	1

טבלת מגמות – כיתות מקצוע (eduMajorsTgrades)

🔑	eduMajorID	Number	מזהה מגמה
🔑	eduTgradeID	Number	מזהה כיתת מקצוע
🔑	eduGradePart	Short Text	שכבה
	eduSchoolID	Number	

eduMajorID	eduTgradeID	eduGradePart	eduSchoolID
20	יא'	10	444737
20	יב'	12	444737
1	י'	13	444737
1	יא'	17	444737
7	יב'	20	444737
1	י'	28	444737
19	יב'	29	444737
1	יב'	30	444737

טבלת משתמשים (eduMembers)

🔑	eduUserID	AutoNumber	מספר המשתמש במערכת (מספור אוטו' - ID(A_))
	eduFirstName	Short Text	השם הפרטי של המשתמש
	eduLastName	Short Text	השם משפחה של המשתמש
	eduPass	Long Text	הסיסמה של המשתמש
	eduType	Short Text	סוג המשתמש
	eduMail	Short Text	אימייל המשתמש (e.g user@ourdomain.com)
	eduID	Short Text	תעודת הזהות הישראלית של המשתמש
	eduGender	Short Text	מגדר המשתמש
	eduGradeID	Number	כיתה/אזור המשתמש (if he is a teacher/admin not applies)
	eduDateRegister	Date/Time	תאריך זמן הרישום של המשתמש
	eduPicture	Short Text	נתיב תמונת המשתמש
	eduBorn	Date/Time	תאריך לידת המשתמש
	eduCityID	Number	עיר התגוררות המשתמש
	eduActive	Short Text	מצב המשתמש
	eduPhone	Short Text	מספר הפלאפון
	eduSchoolID	Number	מזהה בית ספר

eduUs	eduFirstNam	eduLastNam	eduPass	eduType	eduMail	eduID	eduGer	eduGra	eduDateReg	eduPict	eduBorn	eduCityID	eduActive	eduPhone	eduSchoolID
1	אורי	בוגולוק	WBtywUFF6Kh	a	avivbueno@nhswb.org	209280718	m	29	15 11:43:16 AM	/User//Dat	31/7/2003	1615	Yes	058-4084066	444737
149	אביב	בזאם	i3UuSToGcFP74	a	aviv.buen@gmail.com	314711045	m	29	16 03:31:00 PM	/User//Dat	18/3/1997	246	Yes	058-4084066	444737
154	יובל	חזמון	jesCXAVKJF6uS	s	yuvah106@gmail.com	209532498	m	29	16 01:26:56 PM	/User/Data	2/1/1997	246	No	058-4084066	444737
155	אופק	באיזן	3nD0clsmnkZu	a	ofek.avidan@gmail.com	318879574	m	29	16 09:02:54 AM	/IMAGES/p	5/1/1998	152	No	058-4084066	444737
156	יונתן	סינטיסון	cb6LQYmtvpSh	s	scheme101@suicide.it	207001173	m	28	16 07:42:35 PM	/User/Data	1/1/2000	303	No	058-4084066	444737
157	יובל	כחן	UYpHDBQHvNv	s	yuvh129@gmail.com	206375941	m	12	16 10:54:50 PM	/IMAGES/p	10/1/2003	58	No	058-4084066	444737
158	עומרי	ודי	Q0/yznI2FTb61	t	omerivardi@gmail.com	209088590	f	9	16 09:34:32 AM	/User//Dat	10/9/1997	45	Yes	058-4084066	444737
159	מאיר	צוקרבג	kkUrWAC1BbM	t	eti.buen@gmail.com	324940014	f	5	16 07:14:57 PM	/User//Dat	6/10/1970	246	Yes	058-4084066	444737
160	רועי	בזמנן	n5NTIVsfwww	t	roigofman@gmail.com	209282722	m	28	16 02:45:33 PM	/User/Data	6/10/1998	32	Yes	058-4084066	444737
176	שנהב	נחמן	pmIAAHxHHvF	s	yuvsalh106@gmail.com	206396178	m	11	16 09:17:08 AM	/User/Data	11/11/1999	15	Yes	058-4084066	444737
177	יובל	סיוי	HaYLhjeUYEuw	s	MOMO@IAC.COM	206397648	f	4	16 07:02:51 PM	/User/Dat	22/10/1996	776	Yes	058-4084066	444737
178	ירון	מנור	Z8sy3rTUOyYC	s	yarden@gmail.com	206534356	m	3	16 08:04:06 PM	/IMAGES/p	11/11/1999	4	No	058-4084066	444737
179	לירן	ודריגר	BAseIF5+Lws7L	t	a@gmail.com	206575722	f	32	16 09:51:01 AM	/User//Dat	28/12/1989	15	Yes	058-4084066	444737
180	די	שולצי	LUwg0R+9EVNt	s	danimaoi@gmail.com	318849262	m	29	16 09:50:06 AM	/User//Dat	10/11/1999	1616	Yes	058-4084066	444737
183	שחר	קמאי	UXpqz4BvCoXu	s	shahar@gmail.com	318919800	m	29	16 02:17:42 PM	/User//Dat	4/11/1998	1773	Yes	058-4084066	444737
184	ניב	ציון	g67ARJLVE0fJK	s	nivtzi@gmail.com	314952078	m	29	16 02:27:00 PM	/IMAGES/p	15/11/2000	1773	No	058-4084066	444737
185	מנחם	בזן	Y6FQ21tgsDx2	s	eviatarnaim@gmail.com	326626272	m	8	16 05:41:02 PM	/User/Data	9/7/2004	246	Yes	058-4084066	444737
186	עדי	אלטור	CEvdtnry4xuXa	s	testMail@gmail.com	206377020	m	12	17 09:18:15 AM	/Content/g	14/1/1999	1618	Yes	058-4084066	444737
187	אילנה	טוריר	fCxlWBkgf4Ov	s	beni@gmail.com	206364150	f	27	17 08:49:43 AM	/User/Data	14/1/1998	1620	Yes	058-4084066	444737
188	ראם	נגר	3ydoTNWSgcsF	p	aviv.buen@gmail.com	213129075	m	14	17 01:32:40 PM	/User//Dat	30/6/1994	303	Yes	058-4084066	444737
190	דקל	נח	GqCbd6wo06K	s	aviv2@gmail.com	206396186	m	13	17 09:00:38 AM	/User//Dat	4/3/1998	1613	Yes	058-4084066	444737
191	ארחנבן	ישראל	Q6yjj2IF+Wkip	s	ma@gmail.com	054178975	m	28	17 11:34:36 AM	/User/Data	25/3/2003	1681	Yes	058-4084066	444737

טבלת הודעות (eduMessages)

eduMessageID	AutoNumber	מזהה הודעה
eduMessageSender	Number	מזהה משתמש שולח
eduMessageReciver	Number	מזהה משתמש מקבל
eduMessageSubject	Short Text	נושא
eduMessageContent	Long Text	תוכן
eduMessageRead	Yes/No	נקרא/לא נקרא
eduDateSent	Date/Time	תאריך שליחה
eduActive	Short Text	פעיל/לא

eduMessageID	eduMessageSender	eduMessageReciver	eduMessageSubject	eduMessageContent	eduMessageRead	eduDateSen	eduActive
259	149	1	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:16 AM	
260	149	158	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:16 AM	
261	149	159	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input checked="" type="checkbox"/>	17 10:07:16 AM	
262	149	160	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:16 AM	
263	149	176	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:16 AM	
264	149	177	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	:149:
265	149	179	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	
266	149	180	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	
267	149	183	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	
268	149	185	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	
269	149	194	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	
270	149	195	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input checked="" type="checkbox"/>	17 10:07:17 AM	:195:
271	149	186	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	
272	149	187	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	
273	149	188	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	
274	149	190	טיול שנתי לפולין	פגישת הכנה לטיול בפולין ב28/3/2017	<input type="checkbox"/>	17 10:07:17 AM	

טבלת הורה-תלמיד (eduParentMember)

eduUserID	Number	The id of the parent
eduChildID	Number	The id of the child

eduUserID	eduChildID
188	187
188	191

טבלת בתי ספר (eduSchools)

Key	eduSchoolID	Number	The id of the school
	eduSchoolName	Short Text	The name of the school
	eduSchoolLogo	Short Text	The logo of the school
	eduOfficial	Yes/No	Is the school official

	eduSchoolID	eduSchoolName	eduSchoolLogo	eduOfficial
+	112094	יהושע ג		<input checked="" type="checkbox"/>
+	372243	מקיף כאוכב	/User/SchoolData/3d7h2c2c4e3d/image_8.png	<input checked="" type="checkbox"/>
+	444737	שש שנתי נחשון מ.א. חבל מודיעין	/User/SchoolData/4e4e4e7h3d7h/tag1.png	<input checked="" type="checkbox"/>

טבלת ציונים (eduScores)

Key	eduScoreID	AutoNumber	מזהה ציון
	eduStudentID	Number	מזהה משתמש - תלמיד
	eduExamID	Number	מזהה מבחן
	eduScore	Number	ציון

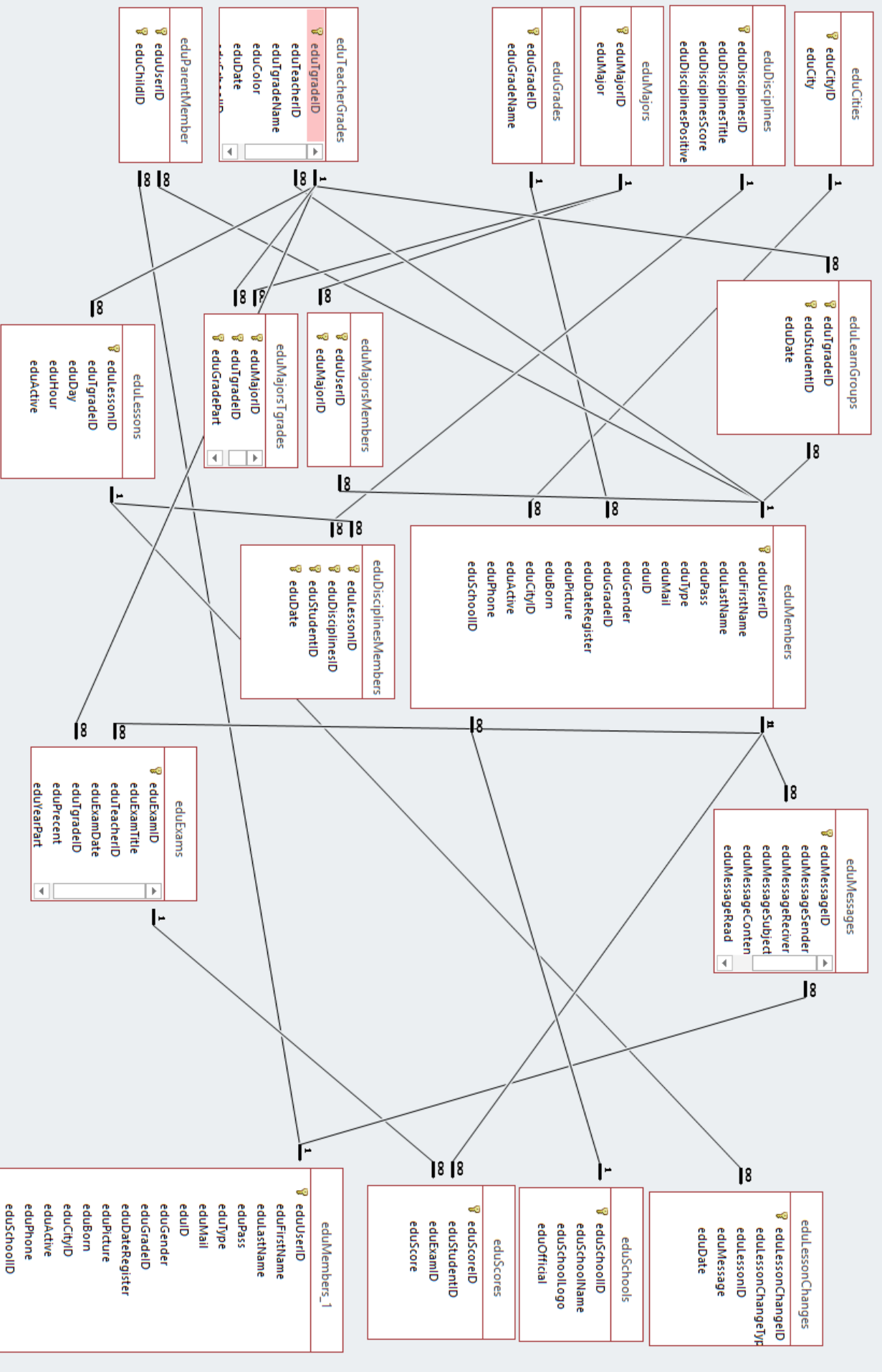
	eduScoreID	eduStudentID	eduExamID	eduScore
	577	157	13	-1
	578	157	22	-1
	579	157	42	-1
	594	178	13	-1
	595	178	22	-1
	596	178	42	-1
	598	191	43	50
	599	184	43	30
	601	191	44	90
	602	184	44	89
	623	187	47	-1

טבלת כיתות מקצוע (eduTeacherGrades)

🔑	eduTgradeID	AutoNumber	מזהה כיתה מקצוע
	eduTeacherID	Number	מזהה משתמש - מורה
	eduTgradeName	Short Text	שם כיתה מקצוע
	eduColor	Short Text	צבע במערכת
	eduDate	Date/Time	תאריך
	eduSchoolID	Number	מזהה בית ספר

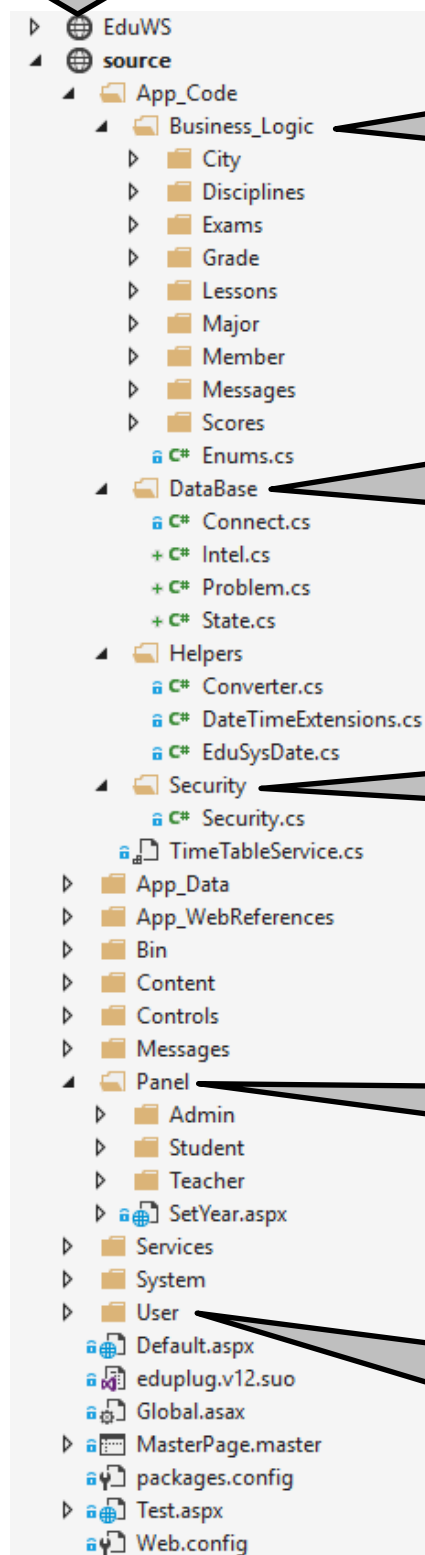
	eduTgradeID	eduTeacherID	eduTgradeName	eduColor	eduDate	eduSchoolID
+	10	159	כיתה בדיקה	2ecc71	1/9/2016	444737
+	11	159	אנגלית 5 יחל	f39c12	1/9/2016	444737
+	12	158	אזרחות	D6CCF7	1/9/2016	444737
+	13	159	יסודות	504DDF	1/9/2016	444737
+	14	159	כיתה מ2015	5E76B1	1/9/2015	444737
+	17	158	כיתה מחשבים קטנה	36F369	1/9/2016	444737
+	20	159	פיזיקה	75A2AF	1/9/2016	444737
+	28	159	תכנות באינטרנט	7FE6C1	1/9/2016	444737
+	29	158	הבנת לקוח	DFE7A7	1/9/2016	444737
+	30	159	שירותי רשת	47A9A0	1/9/2016	444737
+	32	5136	תורה ומקרא	6DD553	1/9/2016	372243

**קשרי
גומלין**



שירות רשת להתאמות

מבנה התיקיות



BI – Business
Intelligence layer

אחראית לכל
הלוגיקה של המערכת

DAL – Data
Access Layer

אחראית לתקשורת
הלוגיקה עם מסדי
הנתונים.

Encryption layer

אחראית להצפנה חזקה
של סיסמאות המשתמשים
טרם כניסתם למסד
הנתונים

Web Interface

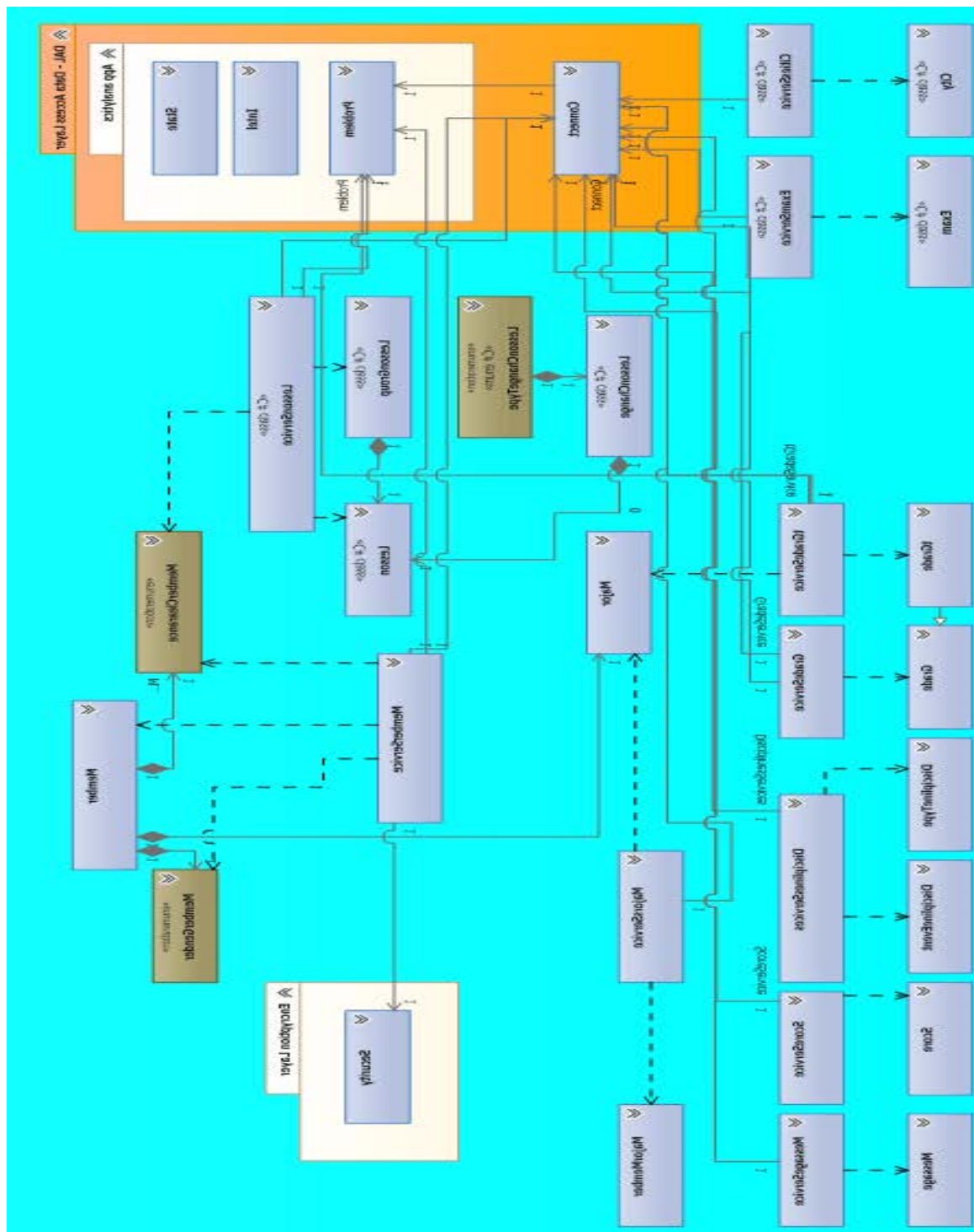
ממשק הWeb של הראשי
של המערכת, מחולק
לשלושה סוגי משתמשים

Web Interface

ממשק
הכניסה/הרשמה/
איפוס סיסמה

המחלקות

UML(tiny.cc/uml_eduplug):



DAL – Data Access Layer:

Connect.cs :

```
/// <summary>
/// This class is connecting the application to the database,
/// mainly used in the business logic classes.
/// </summary>
public static class Connect
{
    /// <summary>
    /// Counts the queries that excuted for the page loaded
    /// </summary>
    public static int QueriesCount;

    /// <summary>
    /// Return's Result For The SQL Query(DataTable Object)
    /// </summary>
    public static DataTable GetData(string SqlQuery, string TableName)

    /// <summary>
    /// Return's Result For The SQL Query(DataSet Object)
    /// </summary>
    public static DataSet GetData(string SqlQuery, string TableName, bool
WantFullDataSet)

    /// <summary>
    /// Return's Object Result (It's Mainly For Mathmatical Use)
    /// </summary>
    public static object GetObject(string SqlQuery)

    /// <summary>
    /// Executes The SqlQuery And Return's Boolean For Success Of The Query Execution
    /// </summary>
    public static bool InsertUpdateDelete(string SqlQuery)

    /// <summary>
    /// Executes The SqlQuery And Return's Boolean For Success Of The Query Execution
    /// </summary>
    public static int InsertUpdateDeleteState(string SqlQuery)
}
```

Intel.cs :

```
/// <summary>
/// Gathers intelligence on client and system, also provides help to UI
/// </summary>
public static class Intel
{
    /// <summary>
    /// Returns the project's root url in order to use the master page in all the web
    directories
    /// </summary>
    /// <returns>Project's root url</returns>
    public static string GetFullRootUrl()

    /// <summary>
    /// Redirects the user to his panel
    /// </summary>
    public static void Redirect()

    /// <summary>
    /// Gets the ip address of the client
    /// </summary>
    /// <returns>String - IP Address of client</returns>
    public static string GetIpAddress()

    /// <summary>
    /// Mark an entry as reviewed
    /// </summary>
    /// <param name="ip">IP of entry</param>
    /// <param name="state">Rev/NoRev</param>
    /// <returns>success</returns>
    public static bool MarkRev(string ip,bool state)

    /// <summary>
    /// Gets all the entries
    /// </summary>
    /// <returns>All the entries</returns>
    public static DataTable GetEduSense()

    /// <summary>
    /// Gets the user device OS
    /// </summary>
    /// <returns>Client device OS</returns>
    public static string GetUserPlatform()

    /// <summary>
    /// Get the current device family of the client
    /// </summary>
    /// <returns>Client Device Family</returns>
    public static string GetDeviceFamily()

    /// <summary>
```

```

    /// Gets the version of the mobile OS of client
    /// </summary>
    /// <param name="userAgent">The user agent of the request</param>
    /// <param name="device">The device of the client</param>
    /// <returns>The mobile version of the device</returns>
    public static string GetMobileVersion(string userAgent, string device)

    /// <summary>
    /// Delete entry
    /// </summary>
    /// <param name="ip">IP</param>
    /// <returns>success</returns>
    public static bool DeleteVisit(string ip)

    /// <summary>
    /// Saves the visitor entry in the DB
    /// </summary>
    public static void SaveVisit()

    /// <summary>
    /// Uses an external API to resolve the current client country by the ip (The API
- IP-API.COM)
    /// </summary>
    /// <returns>Client country</returns>
    public static string GetCountry()

    /// <summary>
    /// Uses an external API to resolve the current client ISP(Internet Service
Provider) by the ip (The API - IP-API.COM)
    /// </summary>
    /// <returns>Client ISP</returns>
    public static string GetISP()

    /// <summary>
    /// Gets entries locations for graph
    /// </summary>
    /// <returns>Dictionary(string, object) of locations</returns>
    public static Dictionary<string, object> GetLocations()

}

```

Problem.cs :

```

    /// <summary>
    /// Logging errors and track to txt files
    /// </summary>
    public static class Problem
    {
        /// <summary>
        /// Log an Exception
        /// </summary>
        /// <param name="ex">Exception occurred in system</param>
        public static void Log(Exception ex)
    }

```

```

    /// <summary>
    /// Save the error to app log
    /// </summary>
    /// <param name="error">Error content</param>
    private static void AddLogApp(string error)

    /// <summary>
    /// Save the error to DB log
    /// </summary>
    /// <param name="error">Error content</param>
    private static void AddLogDB(string error)

    /// <summary>
    /// Save the error to UI log
    /// </summary>
    /// <param name="error">Error content</param>
    private static void AddLogUI(string error)

    /// <summary>
    /// Logs to file
    /// </summary>
    /// <param name="logMessage">Log Message</param>
    /// <param name="w">TextWriter/StreamWriter</param>
    public static void Log(string logMessage, TextWriter w)
}

```

State.cs :

```

    /// <summary>
    /// Deals with txt log
    /// </summary>
    public static class State
    {
        /// <summary>
        /// Logs to file
        /// </summary>
        /// <param name="logMessage">Log Message</param>
        /// <param name="w">TextWriter/StreamWriter</param>
        private static void Log(string logMessage, TextWriter w)
    }

```

BI – Business Intelligence:

Models:

City.cs :

```
/// <summary>
/// City Database Structure
/// </summary>
public class City
{
    /// <summary>
    /// The id of the city
    /// </summary>
    public int ID { get; set; }
    /// <summary>
    /// The name of the city
    /// </summary>
    public string Name { get; set; }
}
```

DisciplineEvent.cs :

```
/// <summary>
/// Discipline Event
/// </summary>
public class DisciplineEvent
{
    /// <summary>
    /// The id of the lesson
    /// </summary>
    public int LessonID { get; set; }
    /// <summary>
    /// The id of the disciplines type
    /// </summary>
    public int DisciplinesID { get; set; }
    /// <summary>
    /// The id of the student
    /// </summary>
    public int StudentID { get; set; }
    /// <summary>
    /// The date of the event
    /// </summary>
    public DateTime Date { get; set; }
}
```

DisciplineType.cs :

```
/// <summary>
/// Discipline Type
/// </summary>
public class DisciplineType
{
    /// <summary>
    /// The name of the type
    /// </summary>
    public string Name { get; set; }
    /// <summary>
    /// The id of the type
    /// </summary>
    public int ID { get; set; }
    /// <summary>
    /// The score
    /// </summary>
    public int Score { get; set; }
}
```

Exam.cs :

```
/// <summary>
/// Exam
/// </summary>
public class Exam
{
    /// <summary>
    /// The id of the exam
    /// </summary>
    public int ID { get; set; }
    /// <summary>
    /// The title of the exam
    /// </summary>
    public string Title { get; set; }
    /// <summary>
    /// The id of the teacher that conducts this exam
    /// </summary>
    public int TeacherID { get; set; }
    /// <summary>
    /// The date of the exam
    /// </summary>
    public DateTime Date { get; set; }
    /// <summary>
    /// The precent of the exam
    /// </summary>
    public int Precent { get; set; }
    /// <summary>
    /// The teacher grade id of exam
}
```

```

    /// </summary>
    public int TeacherGradeID { get; set; }
}

```

Grade.cs :

```

/// <summary>
/// Grade Structure
/// </summary>
public class Grade
{
    /// <summary>
    /// The id of the grade
    /// </summary>
    public int ID { get; set; }
    /// <summary>
    /// The name of the grade
    /// </summary>
    public string Name { get; set; }
}

```

tGrade.cs :

```

/// <summary>
/// Teacher Grade Structure
/// </summary>
public class tGrade : Grade
{
    public int TeacherID { get; set; }
}

```

Lesson.cs :

```

/// <summary>
/// Lesson
/// </summary>
public class Lesson
{
    /// <summary>
    /// The color in the table
    /// </summary>
    public string Color { get; set; }
    /// <summary>
    /// The id of the lesson
    /// </summary>
    public int ID { get; set; }
    /// <summary>
    /// The hour of the lesson 1-12

```



```

    /// </summary>
    public int Hour { get; set; }
    /// <summary>
    /// The day of the lesson 1-6
    /// </summary>
    public int Day { get; set; }
    /// <summary>
    /// The id of the teacher
    /// </summary>
    public int TeacherID { get; set; }
    /// <summary>
    /// The id of the teacher grade
    /// </summary>
    public int TeacherGradeID { get; set; }
    /// <summary>
    /// The name of the grade
    /// </summary>
    public string Name { get; set; }
    /// <summary>
    /// The changes in the system for the lesson
    /// </summary>
    public List<LessonChange> Changes { get; set; }
}

```

LessonChange.cs :

```

/// <summary>
/// Lesson Change
/// </summary>
public class LessonChange
{
    /// <summary>
    /// The id of the lesson change
    /// </summary>
    public int ID { get; set; }
    /// <summary>
    /// The id of the lesson
    /// </summary>
    public int LessonID { get; set; }
    /// <summary>
    /// The date of the change
    /// </summary>
    public DateTime Date { get; set; }
    /// <summary>
    /// The change type
    /// </summary>
    public LessonChangeType ChangeType { get; set; }
    /// <summary>
    /// The message for the change
    /// </summary>
    public string Message { get; set; }
}

```

LessonGroup.cs :

```
/// <summary>
/// LessonGroup
/// </summary>
public class LessonGroup
{
    /// <summary>
    /// The lessons
    /// </summary>
    public List<Lesson> lessons;
    /// <summary>
    /// CTOR
    /// </summary>
    public LessonGroup()
}
```

Major.cs :

```
/// <summary>
/// Major structure
/// </summary>
public class Major
{
    /// <summary>
    /// The id of the major
    /// </summary>
    public int ID { get; set; }
    /// <summary>
    /// The name of the major
    /// </summary>
    public string Title { get; set; }
}
```

MajorMember.cs :

```
/// <summary>
/// Major Member
/// </summary>
public class MajorMember
{
    /// <summary>
    /// The user id of the user
    /// </summary>
    public int UserID { get; set; }
    /// <summary>
    /// The major id
    /// </summary>
    public int MajorID { get; set; }
}
```

Member.cs :

```
/// <summary>
/// Member structure for business logic
/// </summary>
public class Member
{
    //GET/SET
    /// <summary>
    /// The DB id of the user (A_I-PK) - DO NOT ENTER YOURSELF
    /// </summary>
    public int UserID { get; set; }
    /// <summary>
    /// The israeli/passport id of user
    /// </summary>
    public string ID { get; set; }
    /// <summary>
    /// The first name of the user
    /// </summary>
    public string FirstName { get; set; }
    /// <summary>
    /// The last name of the user
    /// </summary>
    public string LastName { get; set; }
    public string Name
    {
        get
        {
            return this.FirstName + ' ' + this.LastName;
        }
        set
        {
            string[] val = value.Split(' ');
            if (val.Length == 2)
            {
                FirstName = val[0];
                LastName = val[1];
            }
        }
    }
}
/// <summary>
/// The E-Mail of the user
/// </summary>
public string Mail { get; set; }
/// <summary>
/// The type of the user - The auth(Admin/Student/Teacher)
/// </summary>
public MemberClearance Auth { get; set; }
/// <summary>
/// The gender of the user
/// </summary>
public MemberGender Gender { get; set; }
/// <summary>
/// The date that the user was born in
```

```

    /// </summary>
    public DateTime BornDate { get; set; }
    /// <summary>
    /// The majors of the user(Computer Science, Movie Director, etc.)
    /// </summary>
    public List<Major> Majors { get; set; }
    /// <summary>
    /// The registration time of the user
    /// </summary>
    public DateTime RegistrationDate { get; set; }
    /// <summary>
    /// The path that the user profile picture is located in
    /// </summary>
    public string PicturePath { get; set; }

    /// <summary>
    /// The id of the user grade
    /// </summary>
    public int GradeID { get; set; }
    /// <summary>
    /// The grade of the user
    /// </summary>
    public Grade Grade { get; set; }
    /// <summary>
    /// The city of the user
    /// </summary>
    public City City { get; set; }
    /// <summary>
    /// The user state active/not active
    /// </summary>
    public bool Active { get; set; }
    //END GET/SET
}

```

Message.cs :

```

    /// <summary>
    /// City Database Structure
    /// </summary>
    public class Message
    {
        /// <summary>
        /// The id of the message
        /// </summary>
        public int ID { get; set; }

        /// <summary>
        /// The id of the reciver
        /// </summary>
        public int ReciverID { get; set; }
        /// <summary>
        /// The id of the sender
        /// </summary>
    }

```

```

public int SenderID { get; set; }
/// <summary>
/// The subject of the message
/// </summary>
public string Subject { get; set; }
/// <summary>
/// The content of the message
/// </summary>
public string Content { get; set; }
/// <summary>
/// The read status of the message
/// </summary>
public bool Read { get; set; }
/// <summary>
/// The sent date
/// </summary>
public DateTime SentDate { get; set; }
/// <summary>
/// State
/// </summary>
public string State { get; set; }
/// <summary>
/// The name of the reciver
/// </summary>
public string ReciverName { get; set; }
/// <summary>
/// The name of the sender
/// </summary>
public string SenderName { get; set; }
/// <summary>
/// Guest
/// </summary>
public bool Guest { get; set; }
}

```

Score.cs :

```

/// <summary>
/// Score
/// </summary>
public class Score
{
    /// <summary>
    /// The id of the score
    /// </summary>
    public int ID { get; set; }
    /// <summary>
    /// The studetn
    /// </summary>
    public Member Student { get; set; }
    /// <summary>
    /// The exam
    /// </summary>
    public Exam Exam { get; set; }
    /// <summary>

```

```

    /// The value of the score
    /// </summary>
    public int ScoreVal { get; set; }
}

```

School.cs :

```

public class School
{
    /// <summary>
    /// The name of the school
    /// </summary>
    public string Name { get; set; }
    /// <summary>
    /// The id of the school
    /// </summary>
    public int Id { get; set; }
    /// <summary>
    /// The logo path of the school
    /// </summary>
    public string LogoPath { get; set; }
    /// <summary>
    /// Is the school officialy recognized by the department of education
    /// </summary>
    public bool Official { get; set; }
    /// <summary>
    /// The school manager
    /// </summary>
    public Member Manager { get; set; }
    /// <summary>
    /// The mid shool manager
    /// </summary>
    public Member MidManger { get; set; }
    /// <summary>
    /// The city of the school
    /// </summary>
    public City City { get; set; }
}

```

Enumerations:

Enums.cs :

```

/// <summary>
/// Enums for business logic
/// </summary>
public enum MemberClearance
{
    Guest = 'g',
    Student = 's',
}

```

```

        Teacher = 't',
        Admin = 'a'
    }
    public enum MemberGender
    {
        Male = 'm',
        Female = 'f',
        Unknown = ''
    }
    public enum LessonChangeType
    {
        Cancel= 'c',
        Fill= 'f',
        Test= 't',
        FinalTest = 'b',
        Unknown = 'u'
    }
}

```

Services:

CitiesService.cs :

```

/// <summary>
/// CitiesService
/// </summary>
public static class CitiesService
{
    /// <summary>
    /// Get all cities
    /// </summary>
    /// <returns>List(City) of all cities</returns>
    public static List<City> GetAll()

    /// <summary>
    /// Get all cities in datatable
    /// </summary>
    /// <returns>DataTable</returns>
    public static DataTable GetAllDT()

    /// <summary>
    /// Get all cities in DataSet
    /// </summary>
    /// <returns>DataSet</returns>
    public static DataSet GetAllDS()

    /// <summary>
    /// Get city by id
    /// </summary>
    /// <param name="cityID">The id of the city</param>
    /// <returns>City</returns>
    public static City GetCity(int cityID)

    /// <summary>
    /// Add a new city

```

```

    /// </summary>
    /// <param name="c">City</param>
    /// <returns>success</returns>
    public static bool Add(City c)

    /// <summary>
    /// Updates city from service
    /// </summary>
    public static void UpdateCities()

    /// <summary>
    /// Updates city from service
    /// </summary>
    private static void updateDB()
}

```

DisciplinesServices.cs :

```

/// <summary>
/// DisciplinesServices
/// </summary>
public static class DisciplinesServices
{
    /// <summary>
    /// Gets all the types
    /// </summary>
    /// <returns></returns>
    public static List<DisciplineType> GetAllTypes()

    /// <summary>
    /// Add new event
    /// </summary>
    /// <param name="lessonID">The id of the lesson</param>
    /// <param name="studentId">The user id of the student</param>
    /// <param name="disciplinesID">The disciplines type id</param>
    /// <param name="date">The date of the event</param>
    /// <returns></returns>
    public static bool Add(int lessonID, int studentId, int disciplinesID,DateTime
date)    /// <summary>
    /// Get all the preselected items
    /// </summary>
    /// <param name="lessonID">The lesson id</param>
    /// <param name="date">The date of the lesson</param>
    /// <returns></returns>
    public static List<DisciplineEvent> GetSelected(int lessonID, DateTime date)

    /// <summary>
    /// Get student disciplines events by user id
    /// </summary>
    /// <param name="uid">User ID</param>
    /// <returns></returns>
    public static DataTable GetStudent(int uid)
}

```



```

    /// <summary>
    /// Get student disciplines events by user id
    /// </summary>
    /// <param name="uid">User ID</param>
    /// <returns></returns>
    public static DataTable GetStudent(int uid,DateTime date)

    /// <summary>
    /// Get student disciplines events by user id
    /// </summary>
    /// <param name="uid">User ID</param>
    /// <returns></returns>
    public static DataTable GetStudent(int uid,int tgid)

    /// <summary>
    /// Reset lesson disciplines
    /// </summary>
    /// <param name="lessonID">LessonID</param>
    /// <param name="date">Date</param>
    public static void ResetLesson(int lessonID, DateTime date)

}

```

ExamService.cs :

```

/// <summary>
/// ExamService
/// </summary>
public static class ExamService
{
    /// <summary>
    /// Get all exams
    /// </summary>
    /// <returns>List of Exams</returns>
    public static List<Exam> GetAll()

    /// <summary>
    /// Add new exam
    /// </summary>
    /// <param name="exm">Exam obj</param>
    /// <param name="tgid">Teacher grade id</param>
    /// <returns>success</returns>
    public static bool Add(Exam exm, int tgid)

    /// <summary>
    /// Get exam by id
    /// </summary>
    /// <param name="eid">Exam id</param>
    /// <returns>Exam</returns>
    public static Exam GetExam(int eid)

    /// <summary>
    /// Get exam id by exam object
    /// </summary>
    /// <param name="exm">Exam</param>

```

```

    /// <returns></returns>
    public static int GetExamID(Exam exm)

    /// <summary>
    /// Delete exam
    /// </summary>
    /// <param name="eid"></param>
    /// <returns></returns>
    public static bool Delete(int eid)

    /// <summary>
    /// Get precent left of teacher grade - irelavent
    /// </summary>
    /// <param name="tgid"></param>
    /// <returns></returns>
    public static int PrecentLeft(int tgid)

    /// <summary>
    /// Get precent left of teacher grade
    /// </summary>
    /// <param name="tgid">teacher grade id</param>
    /// <param name="yearPart">year part</param>
    /// <returns></returns>
    public static int PrecentLeft(int tgid,string yearPart)

    /// <summary>
    /// Get exams by tgrade id
    /// </summary>
    /// <param name="tgid">Teacher grade id</param>
    /// <returns></returns>
    public static List<Exam> GetExamsByTgradeID(int tgid)

    /// <summary>
    /// Get exams by tgrade id
    /// </summary>
    /// <param name="tgid">Teacher grade id</param>
    /// <param name="yearPart">Year part</param>
    /// <returns></returns>
    public static List<Exam> GetExamsByTgradeID(int tgid,string yearPart)

    /// <summary>
    /// Update exam in DB
    /// </summary>
    /// <param name="exam">Exam</param>
    public static bool Update(Exam exam)
}

```

GradesService.cs :

```

/// <summary>
/// GradesService
/// </summary>
public static class GradesService

```

```

{
    /// <summary>
    /// Gets all the grades
    /// </summary>
    /// <returns></returns>
    public static List<Grade> GetAll()

    /// <summary>
    /// Gets all the grades - DataTable
    /// </summary>
    /// <returns>DataTable</returns>
    public static DataTable GetAllDT()

    /// <summary>
    /// Gets all the grades - DataSet
    /// </summary>
    /// <returns>DataSet</returns>
    public static DataSet GetAllDS()

    /// <summary>
    /// Get grade by id
    /// </summary>
    /// <param name="id"></param>
    /// <returns></returns>
    public static Grade Get(int id)
}

```

tGradesService.cs :

```

/// <summary>
/// tGradeService
/// </summary>
public static class tGradeService
{
    /// <summary>
    /// Gets all the tgrades
    /// </summary>
    /// <returns></returns>
    public static List<tGrade> GetAll()

    /// <summary>
    /// Removes tgrade by id
    /// </summary>
    /// <param name="tgid"></param>
    /// <returns></returns>
    public static bool Remove(int tgid)

    /// <summary>
    /// Get id by obj
    /// </summary>
    /// <param name="grd">tGrade OBJ</param>
    /// <returns></returns>
    public static int GetID(tGrade grd)
}

```

```

/// <summary>
/// Add new tGrade to DB
/// </summary>
/// <param name="grd">tGrade</param>
/// <returns></returns>
public static bool Add(tGrade grd)

/// <summary>
/// Add students to tgarde
/// </summary>
/// <param name="tgid">Tgrade ID</param>
/// <param name="students">List of students</param>
/// <returns></returns>
public static bool AddStudents(int tgid, List<Member> students)

/// <summary>
/// Add student to tgarde
/// </summary>
public static bool AddStudent(int tgid, Member student)

/// <summary>
/// Add student to tgarde
/// </summary>
public static bool AddStudent(int tgid, int uid)

/// <summary>
/// Update tgrade
/// </summary>
/// <param name="tg"></param>
/// <returns></returns>
public static bool Update(tGrade tg)

/// <summary>
/// Get part grade of tgarde
/// </summary>
/// <param name="tgid">tgrade id</param>
/// <returns></returns>
public static string GetPartGrade(int tgid)

/// <summary>
/// Get part grade
/// </summary>
/// <param name="gname">grade</param>
/// <returns></returns>
public static string GetPartGrade(string gname)

/// <summary>
/// Get tgrade by id
/// </summary>
/// <param name="tgid">tGrade id</param>
/// <returns></returns>
public static tGrade Get(int tgid)

/// <summary>
/// Get tgrades of teacher

```

```

    /// </summary>
    /// <param name="tid"></param>
    /// <returns></returns>
    public static List<tGrade> GetTeacherTgrades(int tid)

    /// <summary>
    /// Get student count of tgrade
    /// </summary>
    /// <param name="tgid"></param>
    /// <returns></returns>
    public static int GetStudentCount(int tgid)

    /// <summary>
    /// Get students of tgrade
    /// </summary>
    /// <param name="tgid"></param>
    /// <returns></returns>
    public static List<Member> GetStudents(int tgid)

    /// <summary>
    /// Get the major of tgrade
    /// </summary>
    /// <param name="tgid"></param>
    /// <returns></returns>
    public static int GetMajor(int tgid)
}

```

LessonService.cs :

```

    /// <summary>
    /// LessonService
    /// </summary>
    public static class LessonService
    {
        //Days in week
        public static int DaysInWeek =
int.Parse(ConfigurationManager.AppSettings["DaysInWeek"].ToString());
        //Lessons in day
        public static int LessonsInDay =
int.Parse(ConfigurationManager.AppSettings["LessonsInDay"].ToString());
        /// <summary>
        /// Gets all the lessons for the following id.
        /// </summary>
        /// <param name="teacherId">The teacher id</param>
        /// <returns></returns>
        public static List<Lesson> GetAll(int teacherId)

        /// <summary>
        /// Gets a lesson by lesson id
        /// </summary>
        /// <param name="lid">Lesson ID</param>
        /// <returns></returns>
    }

```

```

public static Lesson GetLesson(int lid)

/// <summary>
/// Get all students in lesson
/// </summary>
/// <param name="lid">Lesson id</param>
/// <returns></returns>
public static List<Member> GetAllStudents(int lid)

/// <summary>
/// Helper method
/// </summary>
/// <typeparam name="T"></typeparam>
/// <param name="arrs"></param>
/// <param name="rowIndex"></param>
/// <returns></returns>
private static T[] GetRow<T>(T[,] arrs, int rowIndex)

/// <summary>
/// Gets lessons by teacher
/// </summary>
/// <param name="tid"></param>
/// <returns></returns>
private static DataTable GetLessonsByTeacher(int tid)

/// <summary>
/// Gets lessons by student
/// </summary>
/// <param name="tid"></param>
/// <returns></returns>
private static DataTable GetLessonsByStudent(int sid)

/// <summary>
/// Gets lessons by GradePart
/// </summary>
/// <param name="tid"></param>
/// <returns></returns>
private static DataTable GetLessonsByGradePart(string gradePart)

/// <summary>
/// Gets the time table of the grade that contains this value in its name
/// </summary>
/// <param name="gradePart">Value to check</param>
/// <returns></returns>

/// <summary>
/// Gets the time table of the user
/// </summary>
/// <param name="uid">User ID</param>
/// <param name="memTable">User Clearance</param>
/// <returns></returns>
public static List<LessonGroup[]> GetTimeTable(int uid, MemberClearance memTable)

/// <summary>

```

```

/// Cancel change
/// </summary>
/// <param name="changeID"></param>
/// <returns></returns>
public static bool CancelChange(int changeID)

/// <summary>
/// Get the changes of a lesson
/// </summary>
/// <param name="lid">Lesson id</param>
/// <returns></returns>
public static List<LessonChange> GetChanges(int lid)

/// <summary>
/// Get time table - helper method
/// </summary>
/// <param name="dt">DataTable</param>
/// <returns></returns>
private static List<LessonGroup[]> GetDataTime(DataTable dt)

/// <summary>
/// Gets all the lessons of a tGrade
/// </summary>
/// <param name="tgid">tgrade id</param>
/// <returns></returns>
public static List<Lesson> GetLessons(int tgid)

/// <summary>
/// Delete lesson
/// </summary>
/// <param name="tgid"></param>
/// <param name="day"></param>
/// <param name="hour"></param>
/// <returns></returns>
public static bool DeleteLesson(int tgid, int day, int hour)

/// <summary>
/// Delete lesson
/// </summary>
public static bool DeleteLesson(int lid)

/// <summary>
/// Add lesson to db
/// </summary>
public static bool Add(Lesson lsn)

/// <summary>
/// Add new change to lesson
/// </summary>
/// <param name="change"></param>
/// <returns></returns>
public static bool AddChange(LessonChange change)

/// <summary>
/// Get lesson by params

```

```

    /// </summary>
    /// <param name="tgid"></param>
    /// <param name="hour"></param>
    /// <param name="day"></param>
    /// <returns></returns>
    public static Lesson GetLesson(int tgid,int hour,int day)

}

```

MajorsService.cs :

```

/// <summary>
/// Major Service
/// </summary>
public static class MajorsService
{
    /// <summary>
    /// Get all majors
    /// </summary>
    /// <returns></returns>
    public static List<Major> GetAll()

    /// <summary>
    /// Get all majors - DataTable
    /// </summary>
    /// <returns>DataTable</returns>
    public static DataTable GetAllDT()

    /// <summary>
    /// Get all majors - DataSet
    /// </summary>
    /// <returns>DataSet</returns>
    public static DataSet GetAllDS()

    /// <summary>
    /// Get major by id
    /// </summary>
    /// <param name="majorID">Major id</param>
    /// <returns>Major</returns>
    public static Major Get(int majorID)

    /// <summary>
    /// Get connections of majors and members
    /// </summary>
    /// <returns></returns>
    public static List<MajorMember> GetConnection()

    private static List<Major> currentAll = GetAll();//All
    private static List<MajorMember> currentConnections = GetConnection();//Connections

    /// <summary>
    /// Get majors of user
    /// </summary>
    /// <param name="uid">User ID</param>
    /// <returns></returns>
    public static List<Major> GetUserMajors(int uid)

```



```

    /// <summary>
    /// Update majors and connections with multithreading
    /// </summary>
    /// <returns></returns>
    private static Task Update()

    /// <summary>
    /// Update majors and connections with multithreading
    /// </summary>
    /// <returns></returns>
    private static void updateDB()

    /// <summary>
    /// connect major to tgrade for certain grade part
    /// </summary>
    /// <returns></returns>
    public static bool SetMajorTgrade(int tgid,int mjrid,string gPart)

    /// <summary>
    /// Add new major
    /// </summary>
    /// <param name="m">Major</param>
    /// <returns></returns>
    public static bool Add(Major m)

    /// <summary>
    /// Get major id by name
    /// </summary>
    /// <param name="name"></param>
    /// <returns></returns>
    public static int GetMajorID(string name)
}

```

MembersService.cs :

```

    /// <summary>
    /// A service class for member class
    /// DB Connector
    /// </summary>
    public static class MemberService
    {
        /// <summary>
        /// Get allowed by id
        /// </summary>
        /// <param name="id"></param>
        /// <returns></returns>
        public static Member GetAllowed(string id)

        /// <summary>
        /// Removes the user from the invite list
        /// </summary>
        /// <param name="id">Civilian ID</param>

```

```

    /// <returns></returns>
    public static bool RemoveFromAllowed(string id)
    /// <summary>
    /// Gets all the members from the DB
    /// </summary>
    /// <returns>A list of member object</returns>
    public static List<Member> GetAll()
    /// <summary>
    /// Gets partially field member object for liter work loads that require the
following:
    /// UserID, FirstName, LastName, Name, Auth, Active
    /// </summary>
    /// <returns></returns>
    public static List<Member> GetNames()
    /// <summary>
    /// Gets all the members from the DB
    /// </summary>
    /// <returns>A DataTable of members</returns>
    public static DataTable GetAllDT()
    /// <summary>
    /// Get the current member from the DB
    /// </summary>
    /// <returns>A DataTable of members</returns>
    public static DataTable GetCurrentDT()
    /// <summary>
    /// Login of the user - init for session - The session key is 'Member'
    /// </summary>
    /// <param name="email">Email to login</param>
    /// <param name="pass">Password to login</param>
    /// <returns>Whether the login was successful or not</returns>
    public static bool Login(string email, string pass)
    /// <summary>
    /// Login of the user - init for session - The session key is 'Member'
    /// </summary>
    /// <param name="email">Email to login</param>
    /// <param name="pass">Password to login</param>
    /// <returns>Whether the login was successful or not</returns>
    public static bool Login(string id, string pass, bool ids)
    /// <summary>
    /// Get's the current user from the session
    /// </summary>
    /// <returns>The current logged in user</returns>
    public static Member GetCurrent()
    /// <summary>
    /// Validates Session Keys (Their Values)
    /// </summary>
    /// <param name="sessionNames">The keys</param>
    /// <param name="c">The hyper text transfer protocol context</param>
    /// <returns>Whether the keys are empty or not(if one is empty then it is
false)</returns>
    private static bool ValidateSessions(string[] sessionNames, HttpContext c)
    /// <summary>
    /// Template for inserting a new member into the DB
    /// </summary>

```

```

private const string FullInsertTemplate = "INSERT INTO eduMembers
(eduFirstName,eduLastName,eduPass,eduType,eduMail,eduID,eduGender,eduGradeID,eduPictu
re,eduBorn,eduDateRegister,eduCityID,eduActive)
VALUES('{0}','{1}','{2}','{3}','{4}','{5}','{6}','{7}','{8}','{9}','{10}',{11},Yes)";
/// <summary>
/// Adds a user - ALL FIELDS MUST NOT BE EMPTY!
/// </summary>
/// <param name="m">The member to add</param>
/// <param name="pass">The password of the member</param>
public static void Add(Member m, string pass)
/// <summary>
/// Update the DB allowed table to active account -- just for tracking
/// </summary>
/// <param name="uid">The user id<
/// /param>
public static void UpdateAllowed(int uid)
/// <summary>
/// Updates the session to the following user
/// </summary>
/// <param name="m"></param>
public static void UpdateCurrent(Member m)
/// <summary>
/// Checks if the user is allowed to register - to prevent unwanted guests from
registering
/// </summary>
/// <param name="fname">First Name</param>
/// <param name="lname">Last Name</param>
/// <param name="iuid">Israeli ID</param>
/// <returns>UserAllowed</returns>
public static bool IsAllowed(string fname, string lname, string iuid)
/// <summary>
/// Check if allowed exsist
/// </summary>
/// <param name="id"></param>
/// <returns></returns>
public static bool ExsitsAllowed(string id)
/// <summary>
/// Gets the user auth
/// </summary>
/// <param name="uid">The user id</param>
/// <returns></returns>
public static MemberClearance GetClearance(string uid)
/// <summary>
/// Gets the user id with the id of the user
/// </summary>
/// <param name="ID"></param>
/// <returns></returns>
public static int GetUID(string ID)
/// <summary>
/// Checks if the email already exsits in the database
/// </summary>
/// <param name="email">Email</param>
/// <returns>Whether the member exsist with the same email</returns>
public static bool Exsits(string email)
/// <summary>

```

```

/// Updates the user in the database - USER ID IS MUST
/// </summary>
/// <param name="m">The user</param>
/// <returns>State</returns>
public static bool Update(Member m)
/// <summary>
/// Updates the user password in the database - USER ID IS MUST - PK
/// </summary>
/// <param name="uid">The user id</param>
/// <param name="pass">The new password</param>
/// <returns>State</returns>
public static bool UpdatePassword(int uid, string pass)
/// <summary>
/// Removes the user by the user id PK
/// </summary>
/// <param name="uid">User ID</param>
/// <returns>Action State</returns>
public static bool RemoveFromActive(string uid)
/// <summary>
/// Gets a user by his user id PK
/// </summary>
/// <param name="uid"></param>
/// <returns></returns>
public static Member GetUser(int uid)
/// <summary>
/// Gets a user by his user id PK
/// </summary>
/// <param name="uid"></param>
/// <returns></returns>
public static Member GetUserPart(int uid)
/// <summary>
/// Logs out the current connected user
/// </summary>
public static void Logout()
/// <summary>
/// Returns a list of currently connected users
/// </summary>
/// <returns></returns>
public static List<Member> GetConnected()
/// <summary>
/// Adds a member to invite list(those who are allowed to register)
/// </summary>
/// <param name="m">Member to add</param>
/// <returns></returns>
public static bool AddAllowed(Member m)
/// <summary>
/// Adds a member to invite list(those who are allowed to register)
/// </summary>
/// <param name="fname">First Name</param>
/// <param name="lname">Last Name</param>
/// <param name="id">ID</param>
/// <param name="type">Clearence</param>
/// <returns></returns>
public static bool AddAllowed(string fname, string lname, string id, string type)
/// <summary>

```

```

/// Adds a member to invite list(those who are allowed to register)
/// </summary>
/// <param name="dt"></param>
/// <param name="indID"></param>
/// <param name="indFname"></param>
/// <param name="indLname"></param>
/// <returns></returns>
public static int AddAllowed(DataTable dt,int indID,int indFname,int indLname)
/// <summary>
/// Gets all the users from the following grade
/// </summary>
/// <param name="gid"></param>
/// <returns></returns>
public static List<Member> GetGrade(int gid)
/// <summary>
/// Gets all the users that in that grade incl. Teachers
/// </summary>
/// <param name="grade"></param>
/// <returns></returns>
public static List<Member> GetGradePart(string grade)
/// <summary>
/// Gets all the allowed to register list
/// </summary>
/// <returns>DataTable</returns>
public static DataTable GetAllowed()
/// <summary>
/// Gets the greeting for the panel
/// </summary>
/// <param name="m">Member</param>
/// <returns>Greeting for the panel</returns>
public static string GetGreeting(Member m)
/// <summary>
/// Gets the greeting for the panel
/// </summary>
/// <param name="auth">MemberClearance</param>
/// <param name="gen">MemberGender</param>
/// <returns>Greeting for the panel</returns>
public static string GetGreeting(MemberClearance auth,MemberGender gen)
/// <summary>
/// Get free hours of a teacher
/// </summary>
/// <param name="teacherId"></param>
/// <param name="day"></param>
/// <returns></returns>
public static List<int> GetFreeHours(int teacherId,int day)
/// <summary>
/// Get students of a teacher
/// </summary>
/// <param name="tid"></param>
/// <returns></returns>
public static DataTable GetStudents(int tid)
}

```

MessagesService.cs :

```
/// <summary>
/// Message Service
/// </summary>
public static class MessagesService
{
    /// <summary>
    /// Sends a message
    /// </summary>
    /// <param name="m">Message Object</param>
    public static void SendMessage(Message m)

    /// <summary>
    /// Marks a message as read by message id
    /// </summary>
    /// <param name="mid">Message ID</param>
    public static void MarkAsRead(int mid)

    /// <summary>
    /// Marks a message as read by message object(id)
    /// </summary>
    /// <param name="mid">Message</param>
    public static void MarkAsRead(Message m) { MarkAsRead(m.ID); }

    /// <summary>
    /// Gets the user messages
    /// </summary>
    /// <param name="uid">User ID</param>
    public static List<Message> GetAllUser(int uid)

    /// <summary>
    /// Gets all messages
    /// </summary>
    public static List<Message> GetAll()

    /// <summary>
    /// Deletes a message
    /// </summary>
    /// <param name="uid">User ID</param>
    /// <param name="mid">Message ID</param>
    public static void Delete(int uid, int mid)

    /// <summary>
    /// Gets the unread count of the user
    /// </summary>
    /// <param name="uid">User ID</param>
    public static int GetUnreadCount(int uid)

    /// <summary>
    /// Get unread messages of user
    /// </summary>
    /// <param name="uid"></param>
    /// <returns></returns>
    public static List<Message> GetUnread(int uid)
```

```
}
```

ScoreService.cs :

```
/// <summary>
// ScoreService
/// </summary>
public static class ScoreService
{
    /// <summary>
    /// Get all scores
    /// </summary>
    /// <returns></returns>
    public static List<Score> GetAll()

    /// <summary>
    /// Get all scores of student
    /// </summary>
    /// <param name="sid"></param>
    /// <returns></returns>
    public static List<Score> GetAllStudent(int sid)

    /// <summary>
    /// Get all scores of student in specific tgrade
    /// </summary>
    /// <param name="sid"></param>
    /// <param name="tgid"></param>
    /// <returns></returns>
    public static List<Score> GetAllStudent(int sid,int tgid)

    /// <summary>
    /// Get all scores of student in specific tgrade and year part
    /// </summary>
    /// <param name="sid"></param>
    /// <param name="tgid"></param>
    /// <param name="yearPart"></param>
    /// <returns></returns>
    public static List<Score> GetAllStudent(int sid, int tgid,string yearPart)

    /// <summary>
    /// Get all scores of student in specific tgrade with empty scores
    /// </summary>
    /// <param name="sid"></param>
    /// <param name="tgid"></param>
    /// <returns></returns>
    public static List<Score> GetAllStudentWE(int sid,int tgid)    /// <summary>
    /// Get all scores of student in specific tgrade and year part with empty scores
    /// </summary>
    /// <param name="sid"></param>
    /// <param name="tgid"></param>
    /// <param name="yearPart"></param>
    /// <returns></returns>
    public static List<Score> GetAllStudentWE(int sid, int tgid,string yearPart)
```

```

/// <summary>
/// Get all the scores of an exam
/// </summary>
/// <param name="eid"></param>
/// <returns></returns>
public static List<Score> GetAllExam(int eid)

/// <summary>
/// Get score
/// </summary>
/// <param name="sid">Student ID</param>
/// <param name="eid">Exam ID</param>
/// <returns></returns>
public static Score GetScore(int sid, int eid)

/// <summary>
/// Get avg of exam
/// </summary>
/// <param name="eid">ExamID</param>
/// <returns></returns>
public static double GetAvgExam(int eid)

/// <summary>
/// Get all grade - DataTable
/// </summary>
/// <param name="sid"></param>
/// <returns></returns>
public static DataTable GetAllGrade(int sid)

/// <summary>
/// Get student avg
/// </summary>
/// <param name="sid"></param>
/// <returns></returns>
public static double GetStudentAvg(int sid)

/// <summary>
/// Get student avg
/// </summary>
/// <param name="sid"></param>
/// <param name="tgid"></param>
/// <returns></returns>
public static double GetStudentAvg(int sid,int tgid)

/// <summary>
/// Get student avg final
/// </summary>
/// <param name="sid"></param>
/// <param name="tgid"></param>
/// <returns></returns>
public static double GetStudentAvgFinal(int sid, int tgid)

/// <summary>
/// Get student avg final
/// </summary>

```



```

    /// <param name="sid"></param>
    /// <param name="tgid"></param>
    /// <param name="yearPart"></param>
    /// <returns></returns>
    public static double GetStudentAvgFinal(int sid, int tgid, string yearPart)

    /// <summary>
    /// Add new score
    /// </summary>
    /// <param name="score"></param>
    /// <returns></returns>
    public static bool Add(Score score)

    /// <summary>
    /// Check if exsist
    /// </summary>
    /// <param name="sid"></param>
    /// <param name="eid"></param>
    /// <returns></returns>
    public static bool Exsits(int sid, int eid)

    /// <summary>
    /// Reset scores
    /// </summary>
    /// <param name="eid"></param>
    /// <returns></returns>
    public static bool ResetScores(int eid)    /// <summary>

    /// Reset scores student
    /// </summary>
    /// <param name="sid"></param>
    /// <param name="tgid"></param>
    /// <returns></returns>
    public static bool ResetScoresStudent(int sid, int tgid)
}

```

SchoolService.cs :

```

public static class SchoolService
{
    /// <summary>
    /// Gets all schools
    /// </summary>
    /// <returns></returns>
    public static List<School> GetAll()

    /// <summary>
    /// Update school
    /// </summary>
    /// <param name="school"></param>
    /// <returns></returns>
    public static bool Update(School school)
}

```

```

/// <summary>
/// Gets all known schools to israel (last updated 2-5-2017)
/// </summary>
/// <returns></returns>
public static List<School> GetAllKnown()

/// <summary>
/// Gets all known schools to israel (last updated 2-5-2017)
/// </summary>
/// <returns></returns>
public static List<School> GetKnownCity(int cid)

/// <summary>
/// Gets all known schools to israel (last updated 2-5-2017)
/// </summary>
/// <returns></returns>
public static School GetKnownById(int id)

/// <summary>
/// Add new school to db
/// </summary>
/// <param name="school"></param>
/// <returns></returns>
public static bool Add(School school)

private static List<School> okSchools = new List<School>();
/// <summary>
/// Deletes empty schools
/// </summary>
/// <returns></returns>
public static bool HandleDB()

/// <summary>
/// Check if the school exists
/// </summary>
/// <param name="id"></param>
/// <returns></returns>
public static bool Exsits(int id)

/// <summary>
/// Delete schools
/// </summary>
/// <param name="id"></param>
/// <returns></returns>
public static bool Delete(int id)

/// <summary>
/// Get new id for school
/// </summary>
/// <returns></returns>
public static int GetNID()

/// <summary>
/// Get school by id
/// </summary>
/// <param name="id"></param>
/// <returns></returns>

```

```

        public static School GetSchool(int id)
    }

```

Encryption:

```

/// <summary>
/// *****
/// *****Security*****
/// *****TOP SECRET*****
/// *****
/// </summary>
/*
base64://8J2UsfCd1KXwnZSm8J2UsCDwnZSm8J2UsCDwnZSeIPCd1KDwnZSp8J2UnvCd1LDwnZSwIPCd1LHw
nZS18J2UnvCd1LEg8J2UovCd1KvwnZSg8J2Ur/Cd1LbwnZSt8J2UsfCd1LAg8J2UnvCd1KnwnZSpIPCd1K3wn
ZSe8J2UsPCd1LDwnZS08J2UrPCd1K/wnZSh8J2UsCDwnZS08J2UpvCd1LHwnZS1IPCd1J4g8J2UsPCd1LHwnZ
Sv8J2UrPCd1KvwnZSkIPCd1KLwnZSr8J2UoPCd1K/wnZS28J2UrfCd1LHwnZSm8J2UrPCd1Ks= */
public static class Security
{
    public static int SALT_SIZE = 32; //Salt size
    public static int HASH_SIZE = 64; //Hash size
    public static int PBKDF2_TTT = 512; //Hashing Iteration Count

    /// <summary>
    /// Encryption creator
    /// </summary>
    /// <param name="password">string to encrypt</param>
    /// <returns></returns>
    public static string CreateHash(string password)

    private static byte[] PBKDF2(string password, byte[] salt)

    private static bool SlowEqual(byte[] dbHash, byte[] passHash)

    public static bool ValidatePassword(string password, string dbHash)
}

```

EduWS – Internal Web Service:

```

public class EduAdjustmentsService : System.Web.Services.WebService {
    /// <summary>
    /// Gets all the adjustments types
    /// </summary>
    /// <returns>List of adjustments</returns>
    [WebMethod]
    public List<Adjustment> GetAdjustmentsTypes()

    /// <summary>
    /// Gets the adjustment of the student with the following id
    /// </summary>
    /// <param name="id">ID of the student</param>

```

```

    /// <returns>List of adjustments the student have</returns>
    [WebMethod]
    public List<Adjustment> GetAdjustmentsStudent(string id)

    /// <summary>
    /// Add a student to the system with an adjustment
    /// </summary>
    /// <param name="firstName">First Name</param>
    /// <param name="lastName">Last Name</param>
    /// <param name="id">ID</param>
    /// <param name="adid">Adjustment ID</param>
    /// <returns>The result of the action from the system</returns>
    [WebMethod]
    public string AddStudent(string firstName, string lastName, string id, string adid)

    /// <summary>
    /// Adds an adjustment to a student that already exists in the system
    /// </summary>
    /// <param name="id">ID</param>
    /// <param name="adid">Adjustment ID</param>
    /// <returns>The result of the action from the system</returns>
    [WebMethod]
    public string AddAdjustmentToStudent(string id, string adid)

    /// <summary>
    /// Gets the adjustment of the student with the following id - in JSON
    /// </summary>
    /// <param name="id">ID of the student</param>
    /// <returns>List of adjustments the student have</returns>
    [WebMethod]
    public string GetAdjustmentsStudentJSON(string id)

    /// <summary>
    /// Gets all the students from the system
    /// </summary>
    /// <returns>List of students</returns>
    [WebMethod]
    public List<Member> GetAllStudents()

    /// <summary>
    /// Gets all the students from the system - JSON
    /// </summary>
    /// <returns>List of students</returns>
    [WebMethod]
    public string GetAllStudentsJSON()
}

```

Adjustment.cs :

```

    /// <summary>
    /// Adjustment
    /// </summary>
    public class Adjustment
    {

```

```

    /// <summary>
    /// Name of adjustment
    /// </summary>
    public string Name { get; set; }
    /// <summary>
    /// ID of adjustment
    /// </summary>
    public int ID { get; set; }
}

```

AdjustmentService.cs :

```

/// <summary>
/// AdjustmentService
/// </summary>
public static class AdjustmentService
{
    /// <summary>
    /// Get all adjustments
    /// </summary>
    /// <returns></returns>
    public static List<Adjustment> GetAll()

    /// <summary>
    /// Get students adjustments
    /// </summary>
    /// <param name="sid"></param>
    /// <returns></returns>
    public static List<Adjustment> GetStudent(string sid)

    /// <summary>
    /// Add adjustment to student
    /// </summary>
    /// <param name="sid"></param>
    /// <param name="adid"></param>
    /// <returns></returns>
    public static string Add(string sid, int adid)
}

```

מפת האתר

ראשית, באתר קיימים שלושה סוגי הרשאות:



מנהל



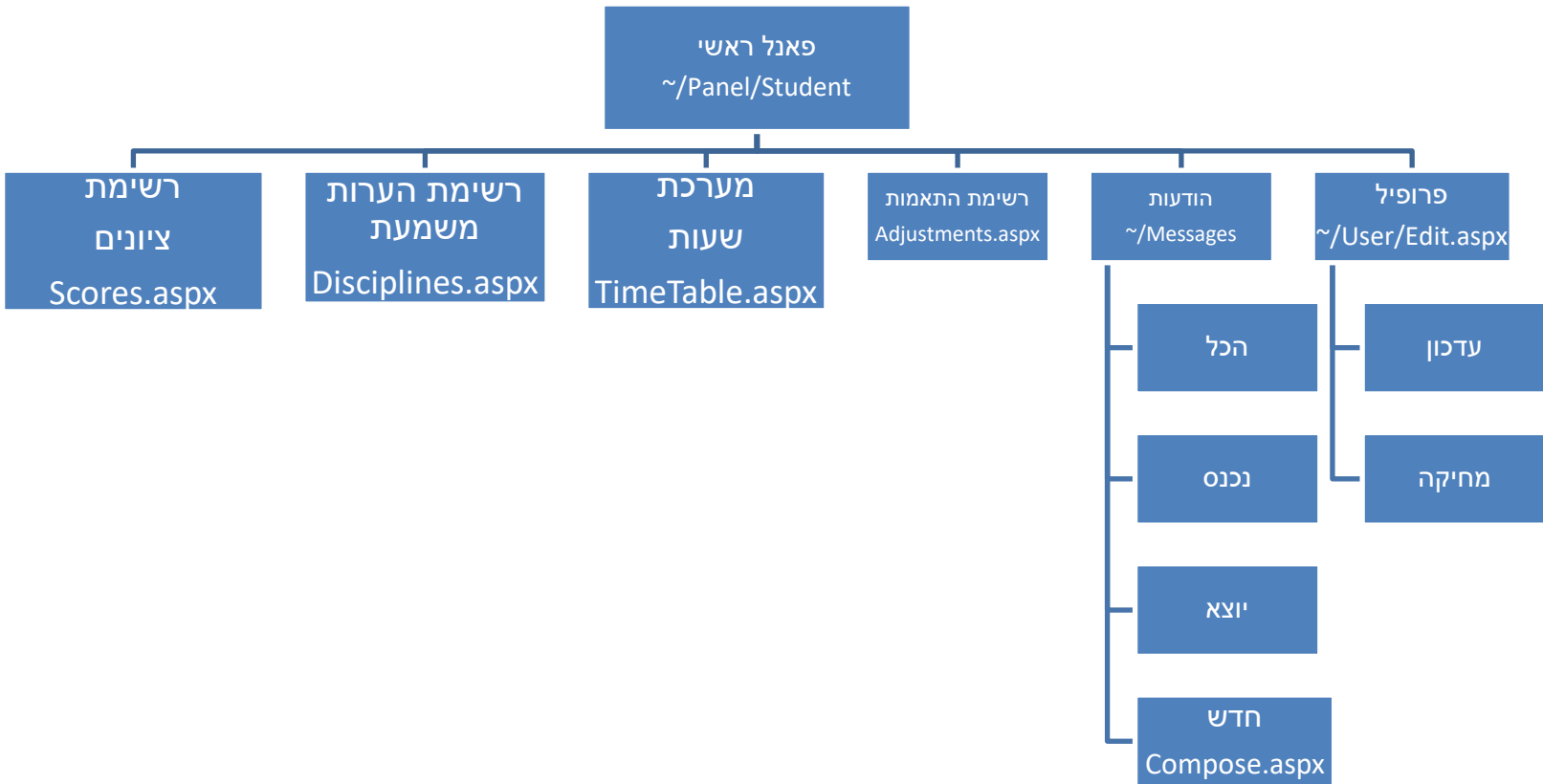
מורה



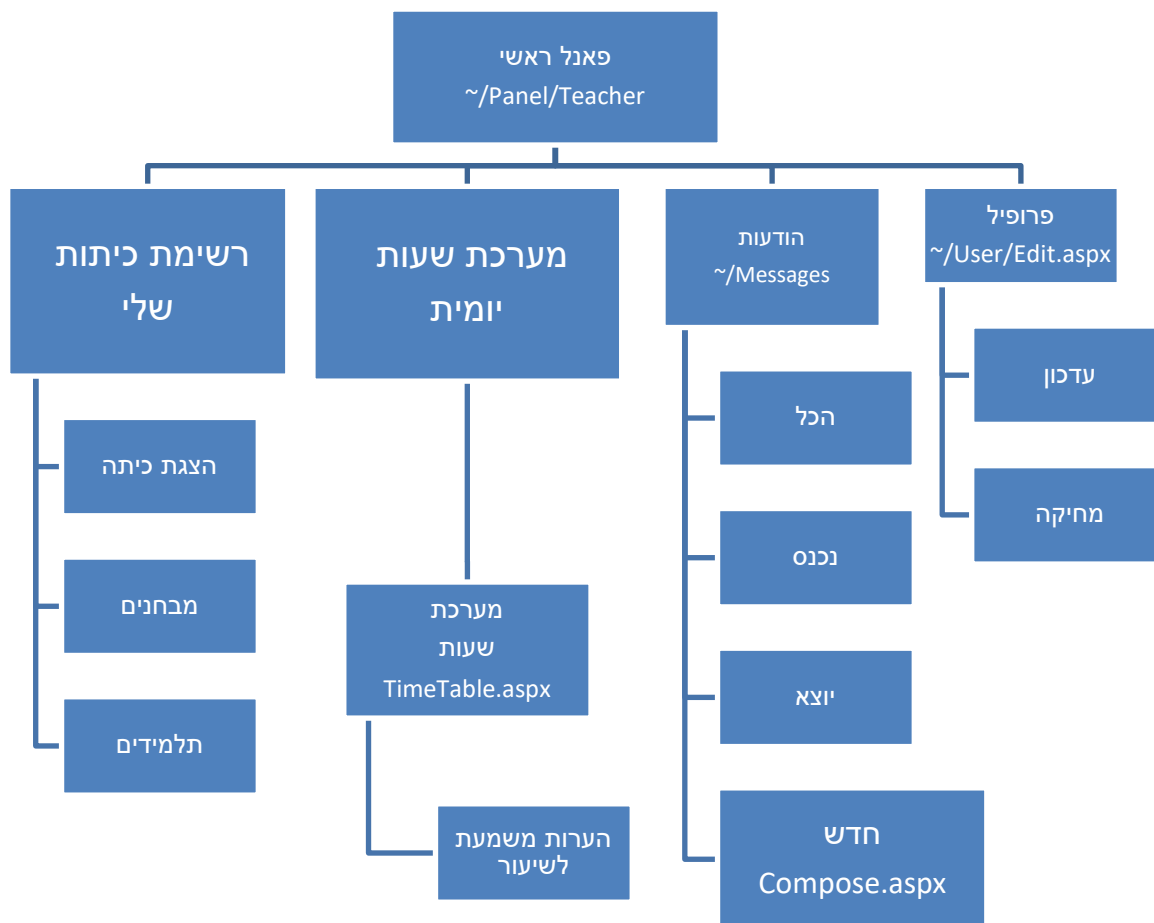
תלמיד/הורה

לכן לכל אחד
מהסוגים מפה
שונה.

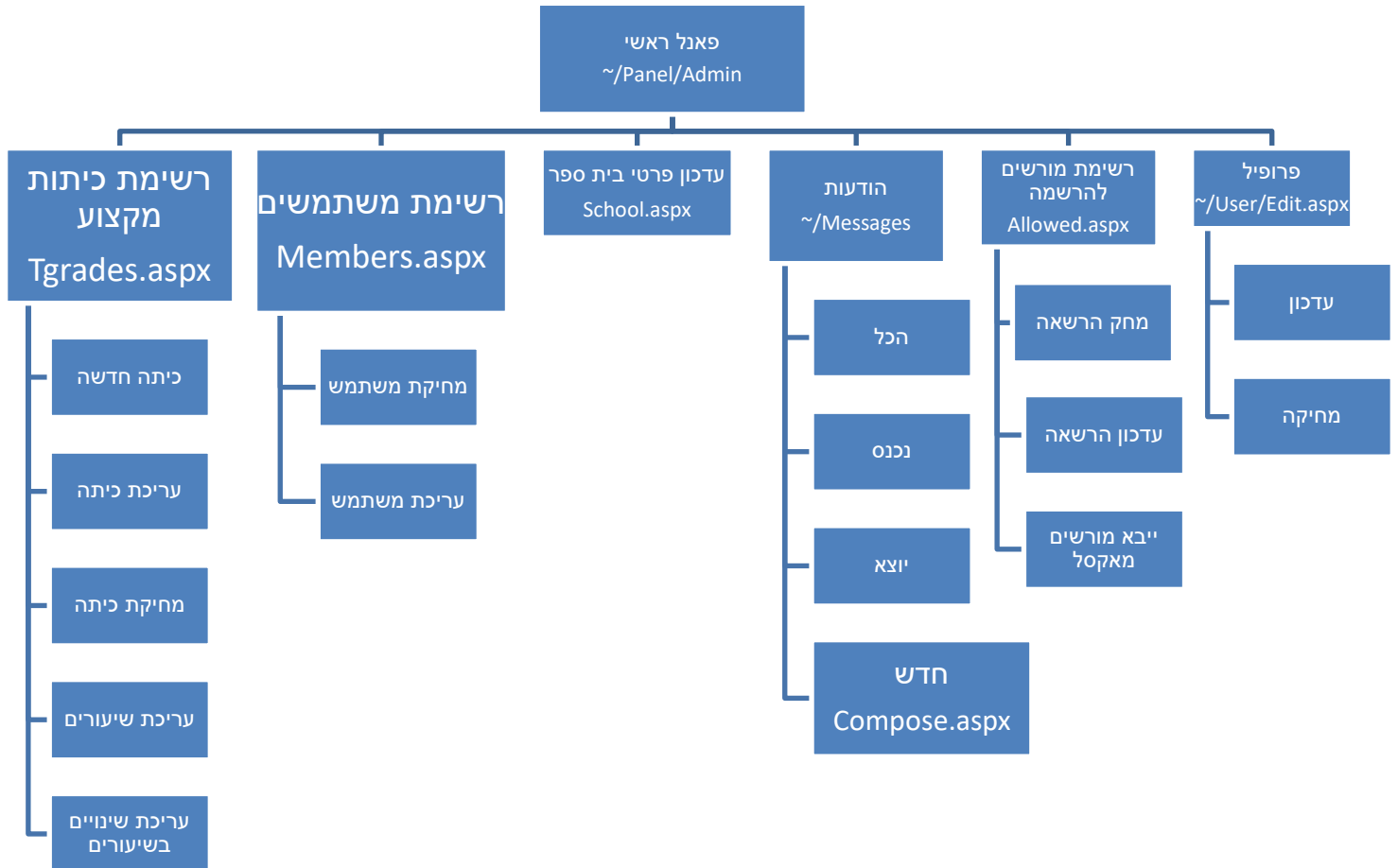
תלמיד:



מורה:



מנהל:



שירותי רשת

בפרויקט זה נעשה שימוש במספר שירותי רשת:

- שירות רשת לייבוא כל הערים בישראל למסד הנתונים:

<http://www.loveburn.com/school/WebServiceCities.aspx>

Web Services Description Language File:

<http://www.loveburn.com/school/WebServiceCities.aspx?wsdl>

- שירות רשת פנימי לניהול ההתאמות של התלמידים.

ביבליוגרפיה / רשימת מקורות / אתרי עזר

- **W3School** – "the world's largest web development site"
<http://www.w3schools.com/>
- **jQuery** – "with less do more" <http://jquery.com/>
- **Codepen.io** - <http://codepen.io/>
- **Material.io** – "Material Design is a unified system that combines theory, resources, and tools for crafting digital experiences." <http://material.io/>
- **Stack Overflow** - <http://stackoverflow.com/>
- **Kudevnet** - <https://www.youtube.com/user/kudvenkat>
- **JQuery DataTables** - <https://datatables.net/>
- **Github.com** – **Important – I used it as a source control for my project.** Find project in <https://github.com/avivbuen/eduplug>

מדריך כללי

כפתור משתמש

פתיחת תפריט

אבא - ראם נגר

התאמות

מערכת שעות

הערות

ציונים

מערכת שעות (05/2017):

אירועי משמעת אחרונים:

אין הערות חדשות

אין ציונים חדשים

כל ההערות

כל הציונים

אבא - ראם נגר

כפתור יציאה

מערכת שעות יומית (לא קיים למנהל)

הגעה לפאנל האישי

מערכת מלאה

מדריך לתלמיד/הורה

אבא - ראם נגר

התאמות

מערכת שעות

הערות

בחירת שנה"ל להצגה

בחירת ילד (במידה והמשתמש הורה)

שש שנות נחשון מא. חבל מוריץ

2017

אורחונץ ישראל

בית

מחיל

הדעות

צא

מערכת מלאה

מדריך למורה

המורה - מארק צוקברג

מערכת שעות מערכת שעות

הודעות

תלמידים

הודעה חדשה +

מערכת שעות (08/05/2017):

כיתות:

1 - כיתה בדיקה

2 - אנגלית 5 יחל

3 - כיתה בדיקה

4 - פיזיקה

5 - חקנות באינטרנט

6 - שירותי רשת

7 - יסודות

8 -

9 -

10 -

11 -

12 -

מערכת מלאה

רשימת כיתות

מעבר לניהול כיתה

המורה - מארק צוקברג

הצגת רשימות חפוש:

תמונה	שם	מייל	מגדר	תאריך לידה	עיר מגורים	פלאפון	פעולות
	שנהב נחמן	yuvsalh106@gmail.com	זכר	11/11/1999	אביאל	058-4084066	
	יובל סיד	MOMO@IAC.COM		22/10/1996	להבים		
	דני שלוצי	danimaol@gmail.com			אביגדור		
	שחר קמאי	shahar@gmail.com			בית נחמיה	058-4084066	
	עדי אלעזר	testMail@gmail.com	זכר	14/01/1999	אביטל	058-4084066	
	אילנה סוטר	beni@gmail.com	נקבה	14/01/1998	אבן יהודה	058-4084066	

חיפוש חופשי

ניתן ללחוץ על עמודה כדי למיין לפיה

אנגלית 5 יחל

תלמידים:	מבחנים:
← אילנה טורנר	← בוחן זמנים
← יונתן סטיבנסון	← סמסטינג שלא עובד
← דני שלוצי	← מבחן ישן
← שחר קמאי	← מבחן (ציונים)


ניהול
מבחן(ציונים)

צור
מבחן
חדש

נתוני
כיתה

מידע על
התלמיד

מדריך למנהל

המנהל - אביב בואנו		הגדרת בית ספר		משתמשים		כיתות		הרשאת הרשמה	
 המנהל - אביב בואנו									
								ממוצע כיתות:	
								57 - כיתה בדיקה 67 - אנגלית 5 יחל לא נקבע - אזרחות לא נקבע - יסודות לא נקבע - כיתת מחשבים קטנה לא נקבע - פיזיקה לא נקבע - תכנות באינטרנט לא נקבע - הבנת לקוח 64 - שירותי רשת <input checked="" type="checkbox"/> כל הכיתות	

ממוצע כיתות

המנהל - אביב בואנו

יבא

רשימת מוזמנים להרשמה

הצגת חשונות

חישוש:

שם	ת.ז.	יבוא	משתמשים	מאקסל	סיסמה זמנית	גיל	פעולות
אבי זרדב	67				4e5f4e7h		
אביב בר	25130243				4e6g0a4e		
אביב סלע	211675657				4e7h9j7h	17.1	
אביב פריד	322889908				4e9j2c2c	16.8	
אביב שחר	322594557				4e8i8i7h	16.9	
אביב אנגלשטיין	323883652				4e8i8i8i	16.9	
אביבוא אברג'ל	213097900				4e7h2c2c	17.3	

המנהל - אביב בואנו

יבא

הוראות להעלת קובץ:

על הקובץ להיות בעל סופס אחד

הקובץ יכיל שלושה עמודות - שם פרטי, שם משפחה, תעודת זהות

הקובץ חייב להיות בפורמט אקסל חדש (xlsx)

סוג:

a- מנהל, ד- מורה, s- תלמיד, ק- הורה

מגדר: זכר/נקבה

מזהי מגמות: מזהי מגמות יופרד באמצעות פסיק (1,2,3) קובץ דוגמא

אלו הם שמות העמודות הדרושים בסדר הבא!

שם פרטי

שם משפחה

תעודת זהות

סוג

כיתה

פלאפון

מגדר

1 - מדעי המחשב

2 - שירותי רשת

3 - ביולוגיה

4 - קולנוע

5 - סוציולוגיה

6 - כימיה

7 - פיזיקה

8 - היסטוריה

1613 - אבסין

1614 - אבסילין

1615 - אביבים

1616 - אביגדור

1617 - אביחיל

1618 - אביטל

1619 - אביעזר

1620 - אבינועם

העלאת קובץ

כאשר יסתיים

תשלח הודעה

{ 62 }

כיתות:

חדש +

הצגת רשומות
חיפוש:

יצירת כיתה
חדשה

חיפוש פתוח

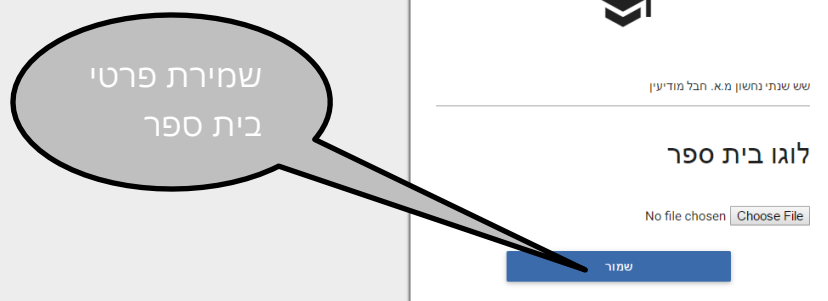
שם מקצוע	מונ	שנינו מערכת	עריכה	שיעורים	מחיקה
אזרחות	עומרי	△			
אנגלית 5 יחל	מארק צוקברג	△			
הבנת לקוח	עומרי ורדי	△			
יסודות	מארק צוקברג	△			
כיתה בדיקה	מארק צוקברג	△			
כיתה מחשבים קסנה	עומרי ורדי	△			

משתמשים

הצגת רשומות
חיפוש:

חיפוש פתוח

תמונה	שם	מייל	תעודת זהות	מגדר	תאריך לידה	תאריך הרשמה	גיל	עיר מגורים	סוג משתמש	שכבה	פעולות
	אורי בוזגלוף	avivbueno@nhswb.org	209280718	זכר	31/07/2003	11:43:16 05/12/2015	13.8	אביבים			
	אביב בואנו	aviv.buen@gmail.com	314711045	זכר	18/03/1997	15:31:00 18/09/2016	20.2	בית אריה	מנהל		
	עומרי ורדי	omerivardi@gmail.com	209088590	זכר	10/09/1997	09:34:32 30/09/2016	19.7	אורי יהודה	מורה	*	
	מארק צוקברג	eti.buen@gmail.com	324940014	נקבה	06/10/1970	19:14:57 03/10/2016	46.6	בית אריה	מורה	*	
	רוני גופמן	roigofman@gmail.com	209282722	זכר	06/10/1998	14:45:33 09/10/2016	18.6	אדמית	מורה	*	



ReadMe

EduPlug - A school managing app

<http://eduplug.co.il>

To setup the project you need to do the following by order:

1. Open the eduplug.sln file on visual studio.
2. Re-add all the web references because visual studio opens different port every time.
3. Run the project.

From here you can use the project in 3 different account types:

Manager:

ID: 314711045

Pass: avivadi1

Teacher:

ID: 324940014

Pass: avivadi1

Student:

ID: 318849262

Pass: avivadi1

Notice: In order to register a new user, you need to first login as a manager and allow this man to register by typing the first name, last name and ID.