

Milestone 2

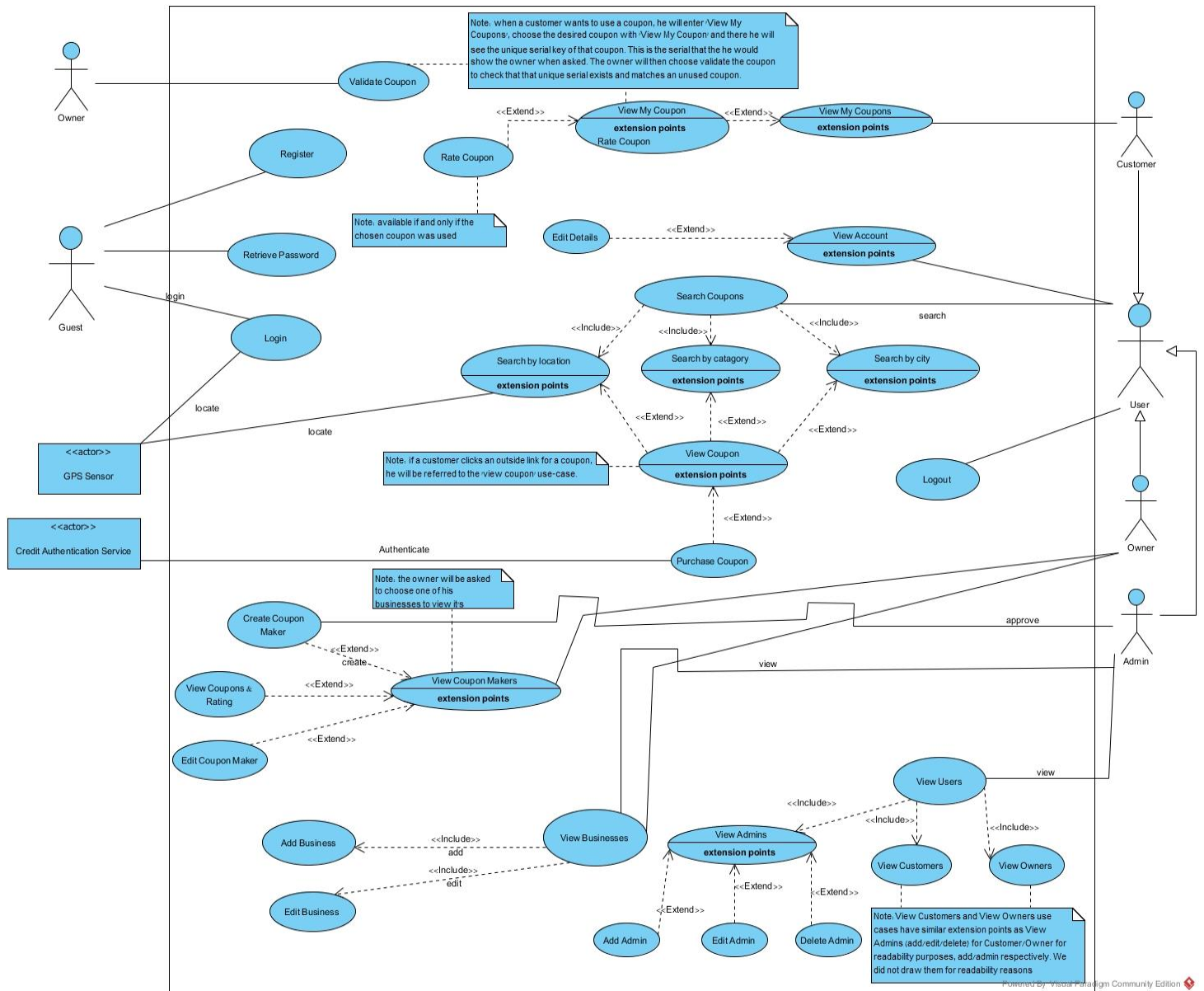
Part A:

Task 1: Changes according to new requirements:

- **Indexing Database:** we decided to add indexes to our database in relevant tables to improve queries runtime (for queries on fields that are often used).
- **Upgrading Server (fictive):** In order to provide the client a server that can support many users with better runtimes for loading pages and running queries, we changed the server's host to high-end top of the line host.
- **Move logic to client-side:** today's computers and smartphones are powerful, in-order to decrease the load from the server (thus increasing its capacity) we chose to move some of the logic to the client-side. In example, if the user submits a weak password (or an empty password) and presses 'register' the data would not be sent to the server, the code at the client-side will notify for a weak password.
- **High Cohesion, Low Coupling:** For a single purpose (in example: edit business), its logic-handling and its view-handling would be separated into different objects (view object & control object).
Also, where needed, a class would offer public static methods so when another class needs to do a calculation that involves objects of the first class, he would not do the calculation himself, but use that function (this way, if the calculation formula is changed, it is transparent to the outside classes). For example: When a user registers he types a password, that password would be encrypted with a key and save in the database. Instead of having multiple functions using the same key to decrypt and encrypt, they would all use the same encrypt decrypt public & static functions offered by the relevant class (This way we could also hide the key for the encryption).
- **Easy new sensor implementation:** In order to add an option for a new sensor, we would write an interface of a sensor, that has one method "sample()" that would return "Sensor Data". For the "Sensor Data": we would add this as an abstract class and for every type of data that could return from a sensor, we will implement a class for that data that would inherit from "Sensor Data" (for example: GPS sensor would implement the interface "Sensor" and therefore the method "sample()", and a new class named "Location" would inherit from "Sensor Data". This way, for a written code, if we want to change the sensor we could just change the line: "Sensor sens = New 'SomeSensor()'" and everything else should stay the same.

Note: requirements document is attached at the end. Updates are marked in green.

Task 2-A: Use-Case Diagram



Note: After consulting with the staff, we decided that "Login" should not be involved in the sequence of each use-case that requires a logged-in user, this is for the reason that once a user is logged in, he does not need to login again to perform a task. Therefore, it is more correct and much clearer to have "Login" as a separate use-case, and have "isLoggedIn()" as a precondition for each required use-case.

Attached in file: **UseCase.jpg**

302991468 אבירם אדירי
308089333 ליהיא ורצ'יק

302491709 ליטל מורלי
303013114 אורי בר-אילן

Permissions Table:

This table will elaborate on specific cases where permissions are needed for an action and the use-case alone does not limit that action to the use of a specific user.

<u>UC \ Actor</u>	<u>Admin</u>	<u>Owner</u>	<u>Customer</u>
Add Business	X		
Purchase Coupons			X
Edit Coupon Maker	X		
Edit Business		X	
"Delete" Business (move to N/A)	X		

Task 2-B: Use Case Scenarios

Use Case #1	
Use case name	Rate Coupon
Description	User will be asked to rate the experience of using a coupon. Rating options will range from 1 to 5 or to not rate at all.
Actors	Client
Pre-conditions	For some customer <i>cust</i> and coupon <i>coup</i> : <code>cust.isLoggedIn()</code> & <code>cust.ownsCoupon(coup.id)</code> & <code>coup.wasUsed()</code> & <code>coup.awaitsRating()</code>
Post-conditions	For some customer <i>cust</i> , coupon <i>coup</i> and integer: <code>coup.awaitsRating() == False</code> & <code>(coup.getRating()) > 0</code> & <code>@before(coup.getNumOfVoters()) + 1 = coup.getNumOfVoters()</code>) <code>(coup.getRating() == 0</code> & <code>@before(coup.getNumOfVoters()) + 1 = coup.getNumOfVoters()</code>)
Main success scenario	View My Coupons> View My Coupon > Rate Coupon 1. Customer decides to rate a coupon that he used sometime before. 2. Customer enters the system. 3. If customer is not logged in: 3.1. User is prompted to log in. 3.2. Reference: Login Use Case: Use Case #3 4. Customer enters "My Coupons" view. 5. Customer chooses the specific coupon that he would like to view. 6. Customer chooses the "Rate" option. 7. Customer chooses a rating (integer from 0 to 5). 8. The coupon is market as rated. 9. The rating of the CouponMaker of this coupon is updated.
Alternative / Extensions	a. If system fails: system would restore the coupon and the coupon maker to their previous state (coupon will be market as unrated and coupon maker's rating will be before the rating). There are no alternatives or extensions for this use case. Of course the user can choose to never use the coupon or use it and never rate it, but there is no actual alternative.

302991468 אבירם אדירי
308089333 ליהיא ורצ'יק

302491709 ליטל מורלי
303013114 אורי בר-אילן

Use Case #2	
Use case name	View Account
Description	User will choose to view his or hers account details.
Actors	User
Pre-conditions	For some user u $u.isLoggedIn()$
Post-conditions	True
Main success scenario	View Account <ol style="list-style-type: none">1. User will decide to view his account details2. If user is logged out:<ol style="list-style-type: none">2.1. User is prompted to login2.2. Reference: Login Use-Case (Use Case #3)3. User enters the "Account" view.
Alternative / Extensions	<ol style="list-style-type: none">a. Edit Details<ol style="list-style-type: none">1. User enters "Edit Details" view.2. User re-enters desired details.3. UI will authenticate correctness (same as sign-up).4. User details will be updated. <p>*. If system fails: user details will be restored to the details before editing.</p>

302991468 אבירם אדירי
308089333 ליהיא ורצ'יק

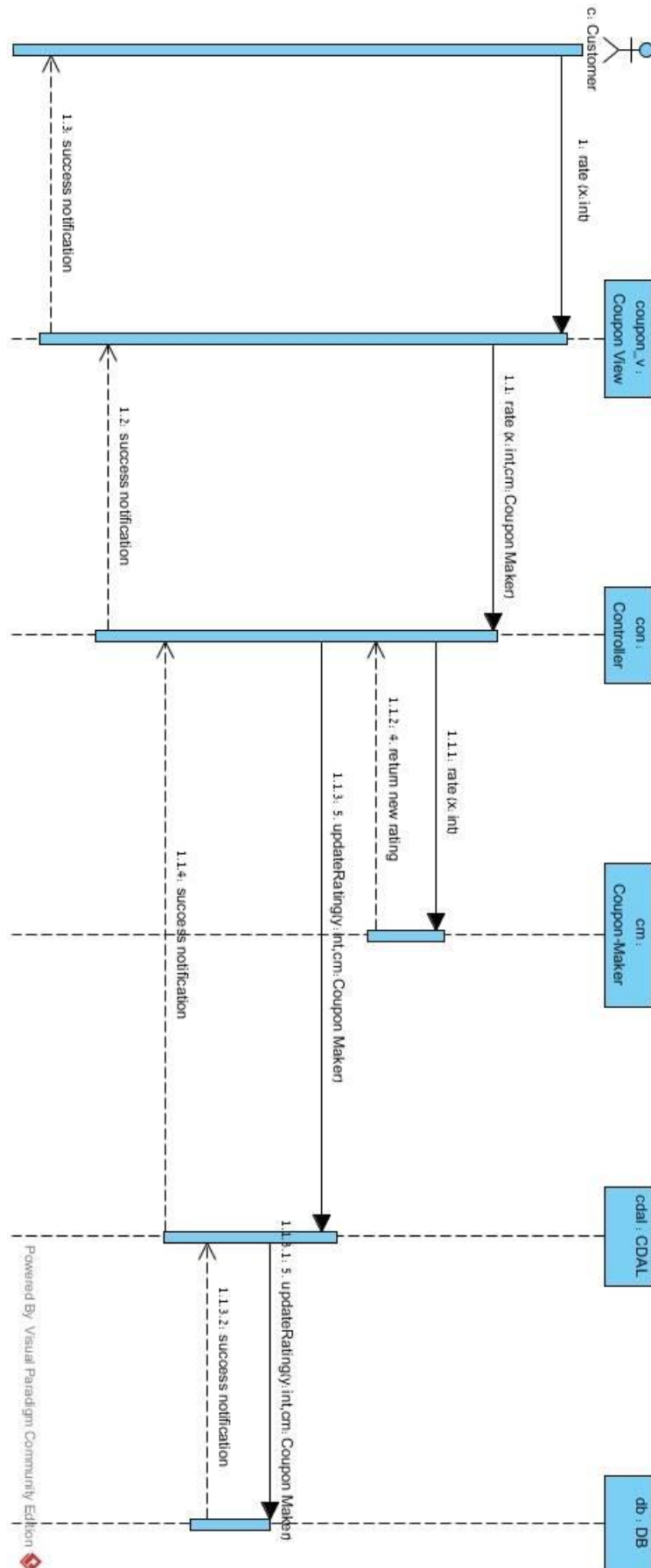
302491709 ליטל מורלי
303013114 אורי בר-אילן

Use Case #3	
Use case name	Login
Description	A guest (an unlogged-in user) will login with his user credentials.
Actors	Guest
Pre-conditions	True
Post-conditions	For some input <i>credentials</i> : If credentials are correct : <code>getUserByCred(credentials)!=NULL</code> else, if they are incorrect: <code>getUserByCred(credentials)==NULL</code>
Main success scenario	Login <ol style="list-style-type: none">1. User decides to login to the system.2. User enters Login view.3. User enters a valid existing username.4. User enters a valid, existing password that is associated to the given username.5. User presses "Login"6. User is now logged in.
Alternative / Extensions	*. If system fails: re-enter login view. A. User entered incorrect details: <ol style="list-style-type: none">1. If user entered incorrect details:<ol style="list-style-type: none">1.1. System will prompt the user to re-enter his data1.2. If user re-enters his credentials and presses "login":<ol style="list-style-type: none">1.2.1. Return to 1.

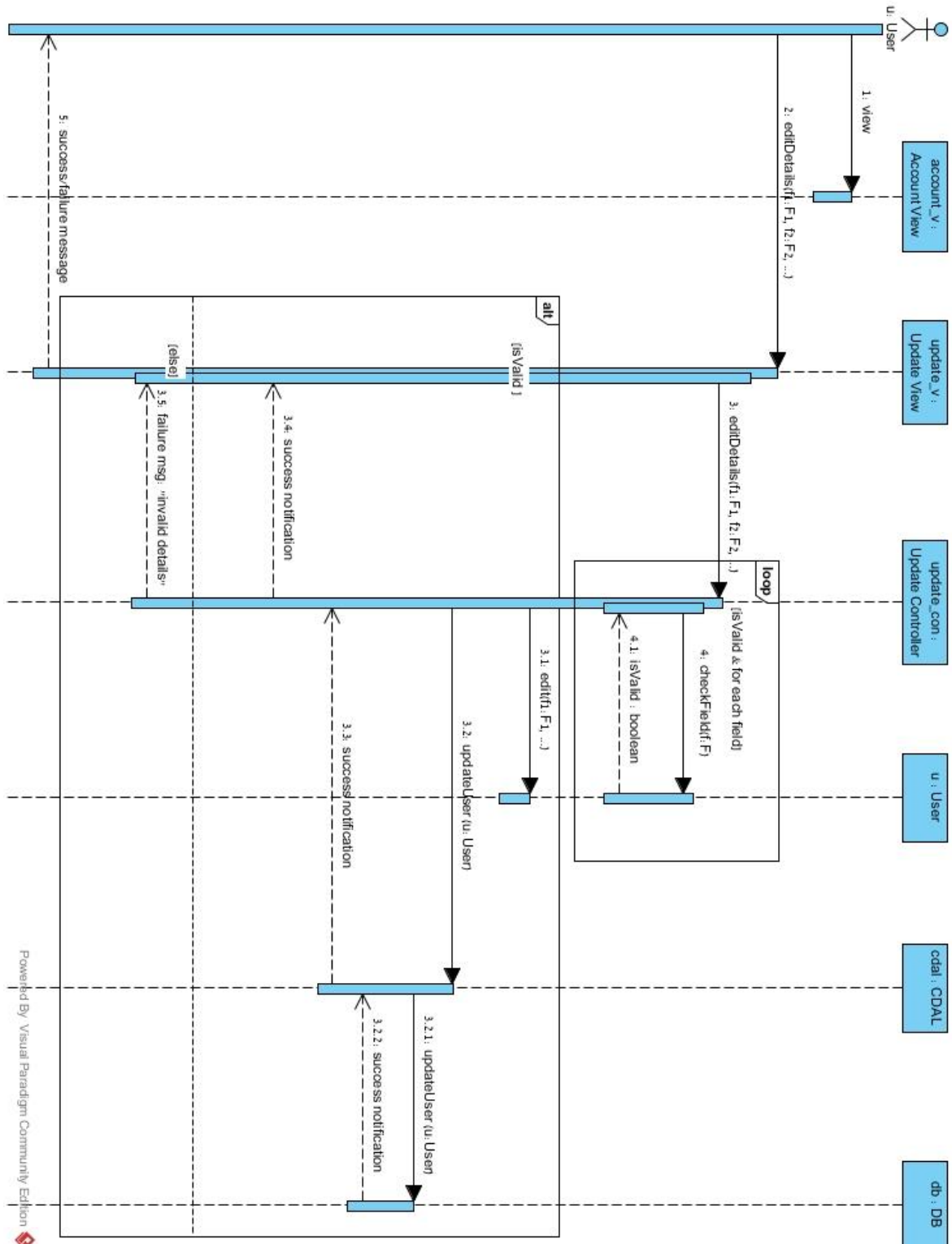
Use Case #4	
Use case name	View Businesses
Description	If user is an owner: he will be able to view his or hers businesses If user is an admin: he will be able to view all existing businesses
Actors	Owner, Admin
Pre-conditions	For some user <i>user</i> : user.isLoggedIn() && (user.isAdmin() user.isOwner()) Note: if user is an owner and does not have business access is still granted but his view will be empty.
Post-conditions	True
Main success scenario	View Businesses <ol style="list-style-type: none">1. User decides to access businesses view2. If user is not logged in<ol style="list-style-type: none">2.1. User is prompted to log-in.2.2. Reference: Login Use-Case (Use Case #3)3. The system shows a list of all relevant businesses (if the user is an owner, all of his/hers businesses, if the user is an admin, all existing businesses).
Alternatives/Extensions	<ol style="list-style-type: none">a. View Business<ol style="list-style-type: none">1. User select a specific business from the view and sees it's details (such as rating, coupons that were purchased, ect.)b. Edit Business (owner view)<ol style="list-style-type: none">1. System verifies that the user is an owner.2. Owner re-enters data that he wishes to change (note that not all of the business data is changeable).3. User presses "save".c. Edit Business (admin view)<ol style="list-style-type: none">1. System verifies that the user is an admin.2. Admin chooses a new state for the business (for example: "inactive").3. User presses "save". <p>*. If system fails on b or c: restore data to its previous state before entering b or c (respectively).</p> <p>Note: for legal purposes, a full delete option is not available (in order to track the business and owner related to a specific coupon). Instead, Admin would be able change the business status to 'inactive' in the edit function.</p>

Task 3: Sequence Diagram

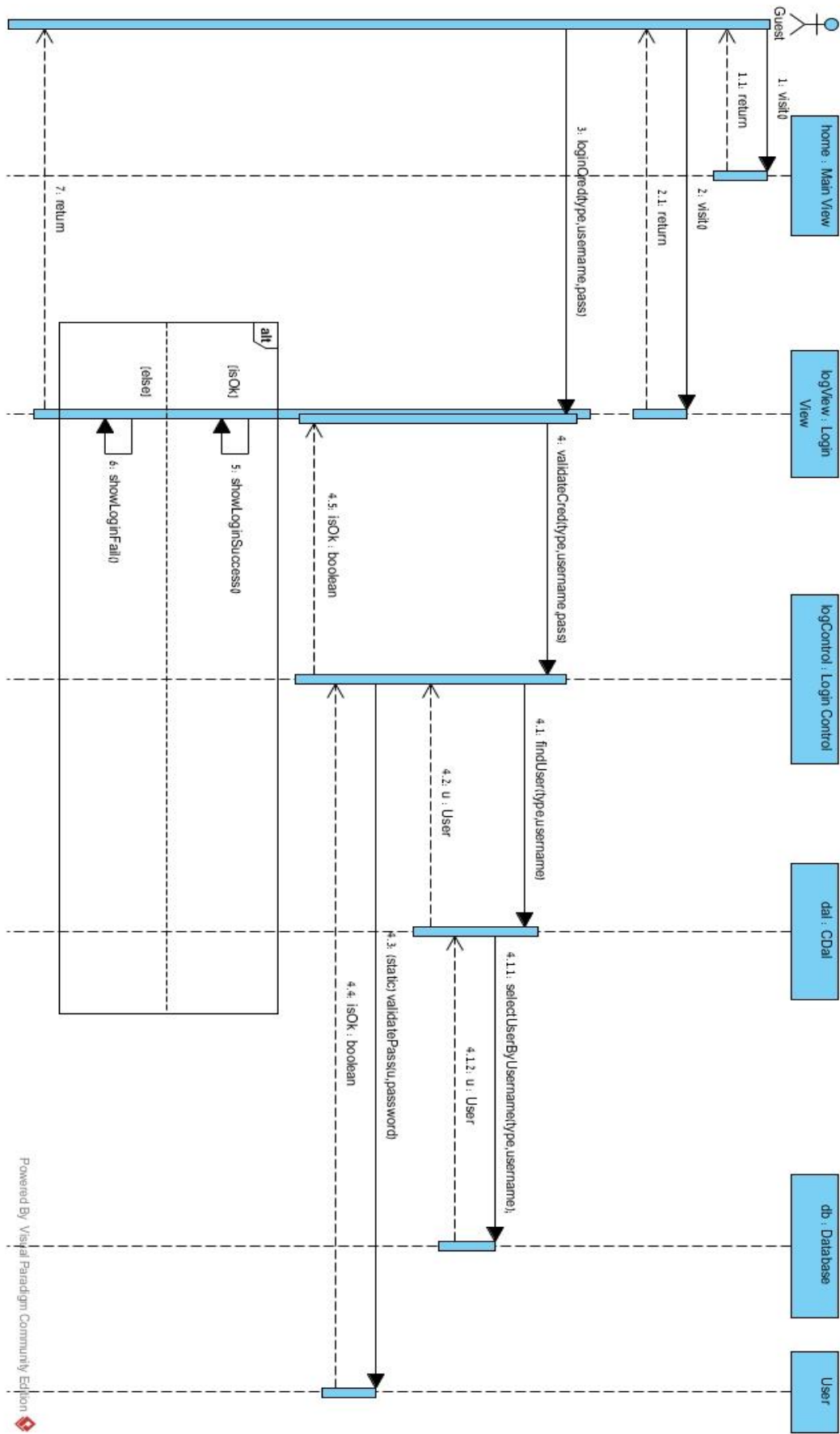
Sequence-UC1-Rate Coupon.jpg:



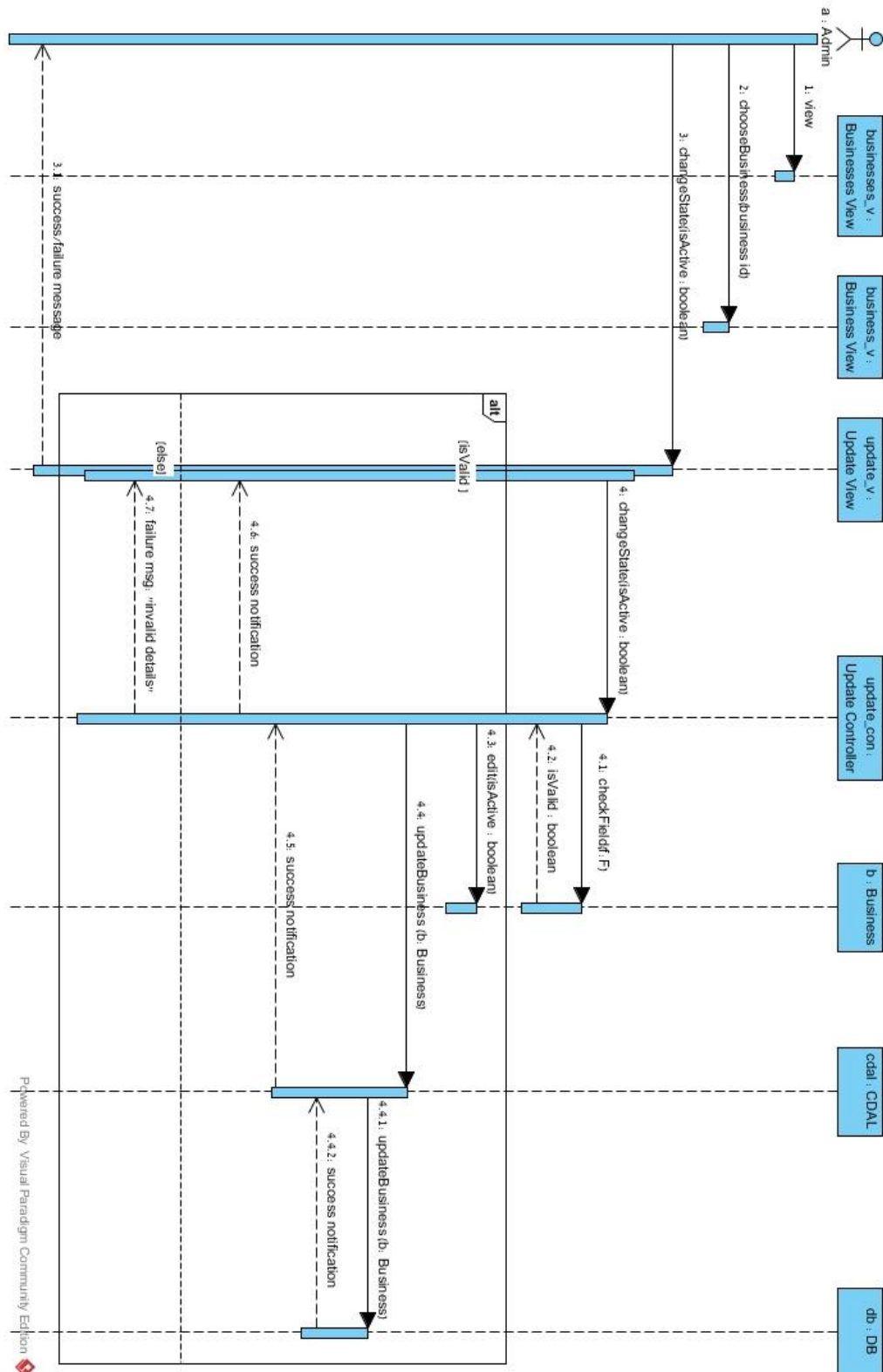
Sequence-UC2-View Account+Edit Details.jpg:



Sequence-UC3-Login.jpg:



Sequence-UC4-View Business+Edit Details.jpg:

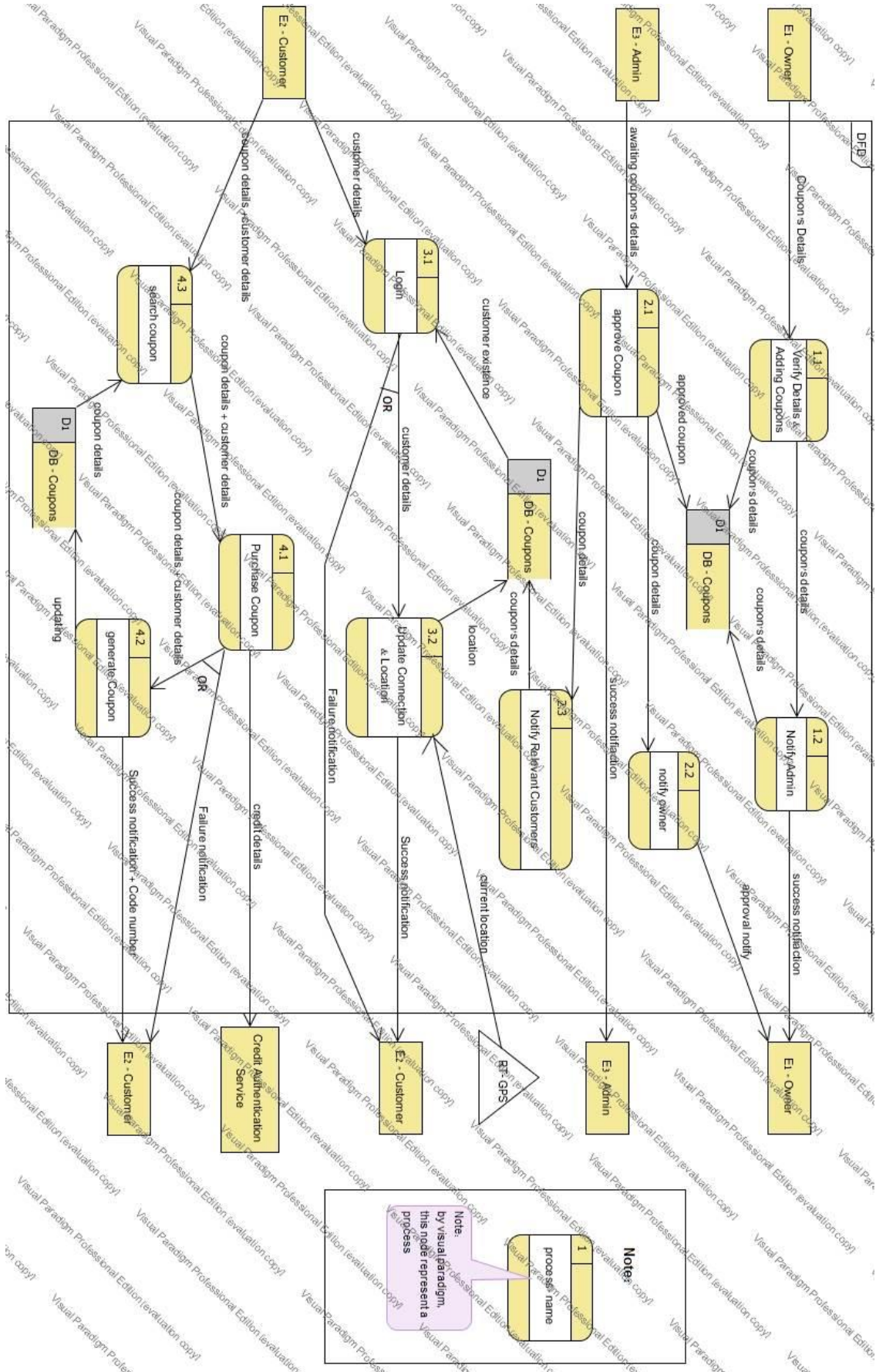


אבירם אדירי 302991468
ליהיא ורצ'יק 308089333

ליטל מורלי 302491709
אורי בר-אילן 303013114

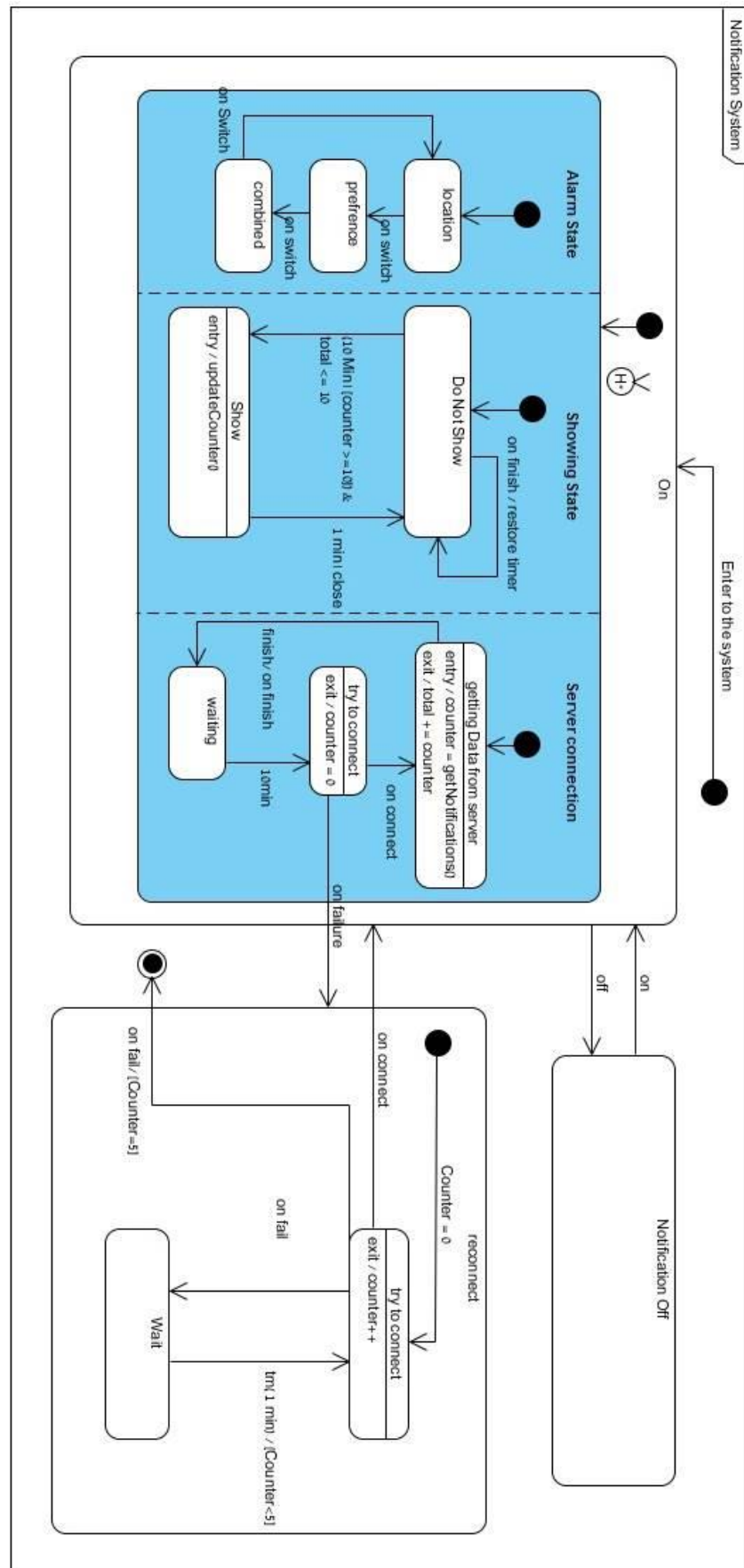
Task 4: DFD

DFD.jpg



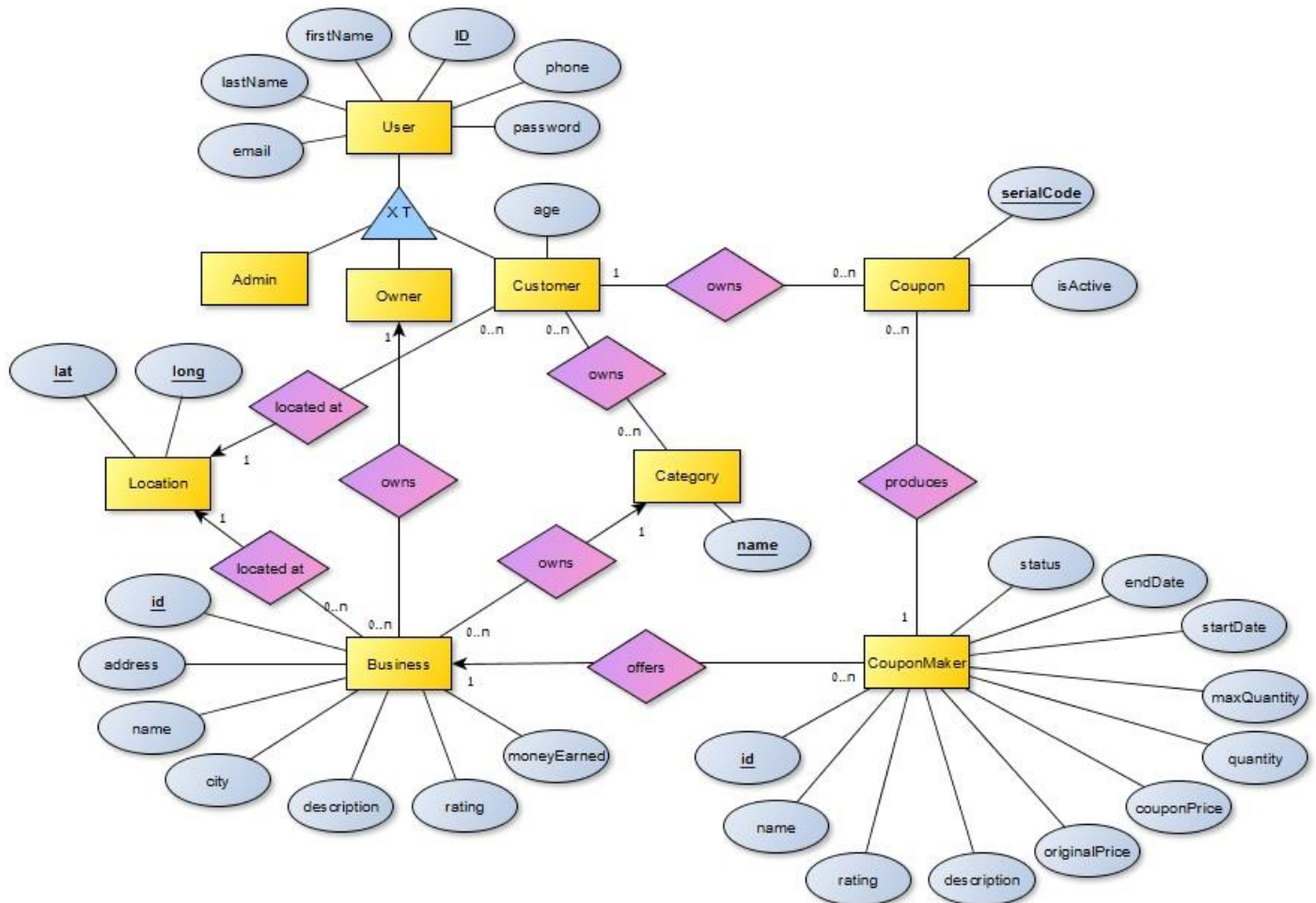
Task 5: State Machine Diagram

[State machine.jpg](#)



Task 6: Updating previous Diagrams

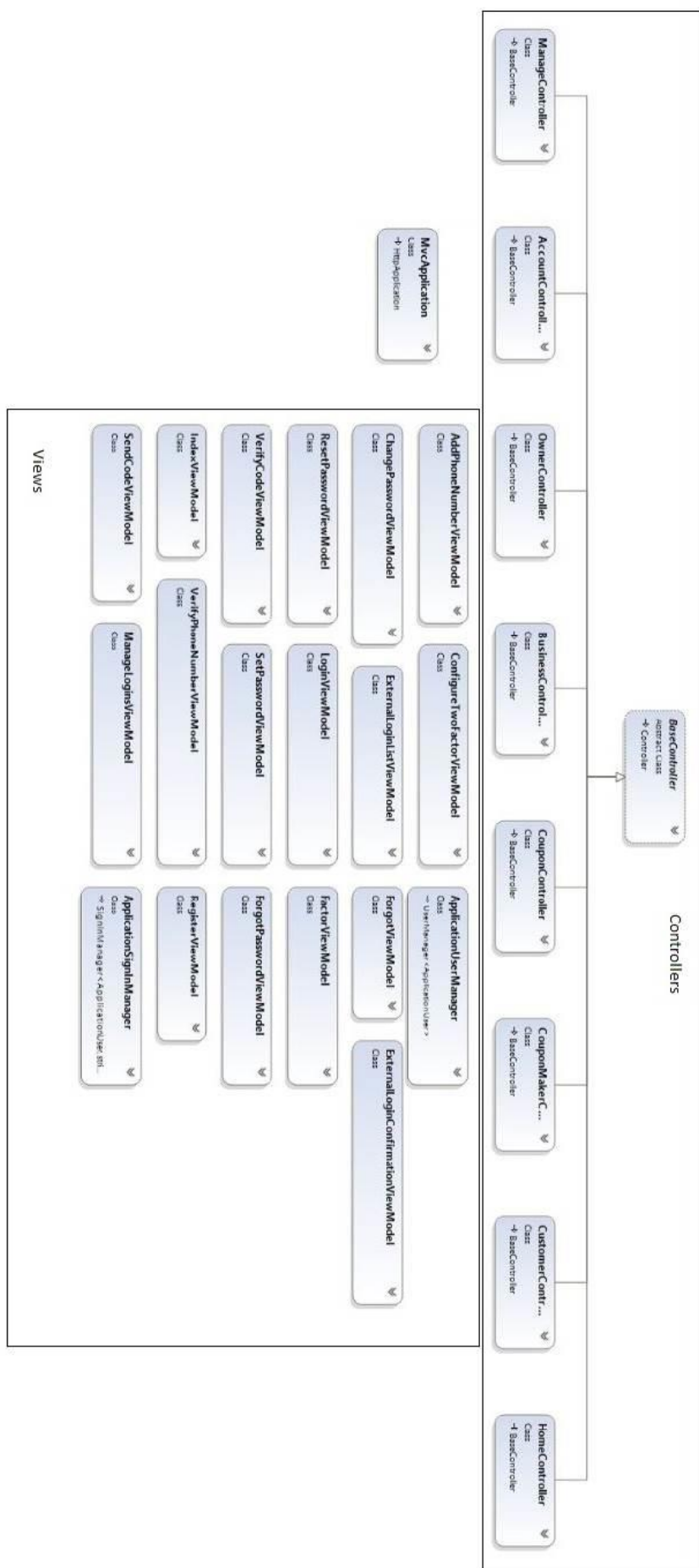
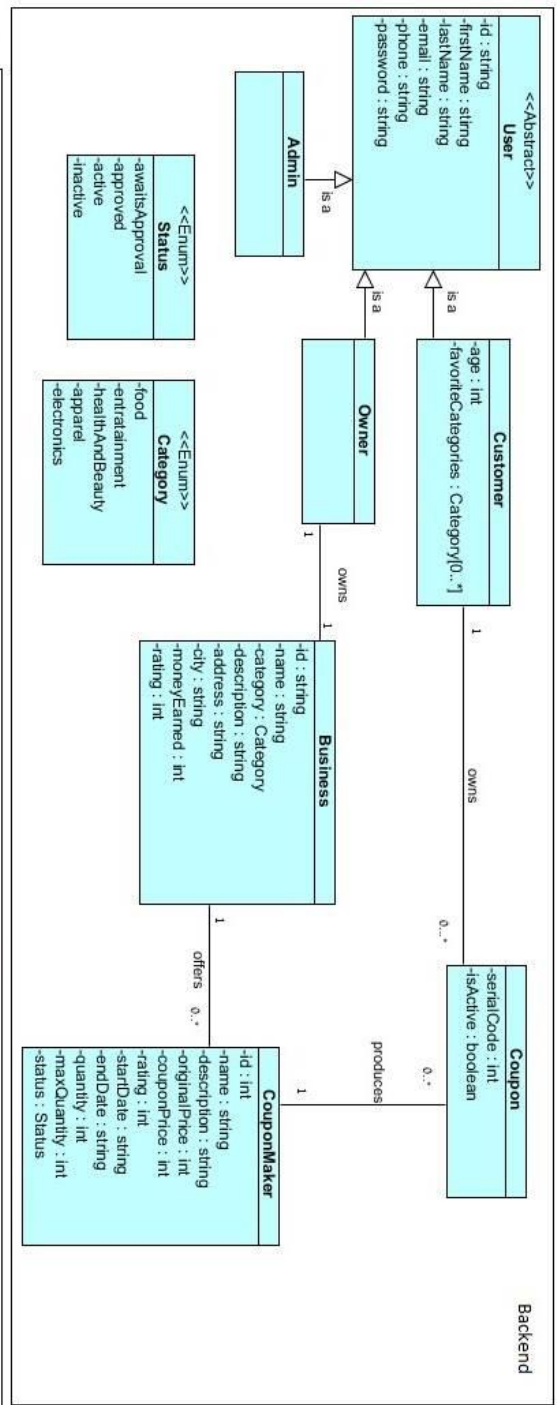
ERD:



אבירם אדירי 302991468
ליהיא ורצ'יק 308089333

ליטל מורלי 302491709
אורי בר-אילן 303013114

UML:



אבירם אדירי 302991468
ליהיא ורצ'יק 308089333

ליטל מורלי 302491709
אורי בר-אילן 303013114

Requirements Document:

מס'ד	קבוצת תהליך	תהליך	תאור הדרישה	מקור הדרישה	סוג	עדיף
מספור חד ערכי	תהליך עסקי	קבוצת תהליכים בעלת מכנה משותף	תיאור כללי של הדרישה	בעל העניין / המסמך ממנו מוצתה הדרישה	פונקציונלית / לא-פונקציונלית	סולם ערכים
1.1.1	ניהול קופונים	הוספת קופון	המערכת תתמוך בהוספת קופון (שם, תיאור, קטגוריה, מחיר מקורי, מחיר לאחר הנחה, דירוג, תאריך אחרון למימוש) עבור בית עסק מסוים.	לקוח	פונקציונלית	5
1.1.2	ניהול קופונים	הוספת קופון	לאחר אישור של מנהל המערכת על הוספת הקופון, המשתמש יקבל התראה על קופונים לאור המיקום ו/או העדפות המשתמש.	לקוח	פונקציונלית	5
1.1.3	ניהול קופונים	הוספת קופון	הוספת קופון תתאפשר ע"י מנהל מערכת ובתי עסק בלבד.	לקוח	לא פונקציונלית	5
1.1.4	ניהול קופונים	הוספת קופון	המערכת תשמור את הקופונים הקיימים במערכת עד לשנה לאחר תאריך המימוש האחרון.	מנתח מערכות	לא פונקציונלית	2
1.1.5	ניהול קופונים	הוספת קופון	הוספת קופון מרשת חברתית תתאפשר ע"י כל משתמש שנרשם למערכת.	לקוח	לא פונקציונלית	3
1.2.1	ניהול קופונים	עריכת פרטי קופון	עריכת פרטי קופון ומחיקתו תתאפשר ע"י מנהל מערכת בלבד.	לקוח	לא פונקציונלית	5
1.3.1	ניהול קופונים	אישור קופון	מנהלי מערכת יאשרו קופונים שהוזנו ע"י משתמשים ובתי עסק.	לקוח	פונקציונלית	4
1.4.1	ניהול קופונים	חיפוש קופון	המערכת תתמוך בחיפוש קופונים ע"י משתמשים ובעלי עסק.	לקוח	פונקציונלית	5
1.4.2	ניהול קופונים	חיפוש קופון	משתמש יוכל לבצע חיפוש של קופון לפי מיקומו וכן לפי בתי עסק וקטגוריות (מסעדות, פנאי וביילי, צרכנות וכו').	לקוח	פונקציונלית	5
1.4.3	ניהול קופונים	חיפוש קופון	חיפוש קופון לפי מיקום יתאפשר באמצעות מפה ולפי מיקום GPS או הזנת עיר.	לקוח	פונקציונלית	4
2.1.1	ניהול מקומות	הוספת בית עסק	המערכת תתמוך בהוספת בתי עסק למימוש קופונים (שם, כתובת, עיר, תיאור, קטגוריה).	לקוח	פונקציונלית	5
2.1.2	ניהול מקומות	הוספת בית עסק	הוספת בית עסק תתאפשר ע"י מנהל מערכת בלבד.	לקוח	לא פונקציונלית	5
2.2.1	ניהול מקומות	חיפוש בית עסק	המערכת תתמוך בחיפוש בתי עסק המעניקים קופונים.	לקוח	פונקציונלית	3
2.2.2	ניהול מקומות	חיפוש בית עסק	משתמש יוכל לבצע חיפוש של בית עסק לפי מיקומו וכן לפי קטגוריות (מסעדות, פנאי וביילי, צרכנות וכו').	לקוח	פונקציונלית	3
2.2.3	ניהול מקומות	חיפוש בית עסק	חיפוש בית עסק לפי מיקום יתאפשר באמצעות מפה ולפי מיקום GPS או הזנת עיר.	לקוח	פונקציונלית	1
2.3.1	ניהול מקומות	עריכת פרטי בית עסק	עריכת פרטי בית העסק תתאפשר ע"י בית העסק בלבד.	לקוח	לא פונקציונלית	4
2.4.1	ניהול מקומות	מחיקת בית עסק	מחיקת בית עסק תתאפשר ע"י מנהל המערכת בלבד.	לקוח	לא פונקציונלית	4
3.1.1	ניהול משתמשים	הוספת משתמש	המערכת תתמוך בהוספת משתמש בעת כניסתו הראשונית למערכת.	לקוח	פונקציונלית	5
3.1.2	ניהול משתמשים	הוספת משתמש	בעת הוספת משתמש, המערכת תדרוש פרטי זהויה (שם משתמש, מייל, טלפון) וסיסמא.	לקוח	מפוצל	5
3.1.2.1	ניהול משתמשים	הוספת משתמש	בעת הוספת משתמש, המערכת תדרוש פרטי זהויה (שם משתמש, מייל, טלפון).	מנתח מערכות	פונקציונלית	5
3.1.2.2	ניהול משתמשים	הוספת משתמש	בעת הוספת משתמש, המערכת תדרוש סיסמא חזקה.	מנתח מערכות	לא פונקציונלית	5
3.1.3	ניהול משתמשים	הוספת משתמש	המשתמש יוכל להירשם למערכת באמצעות מכשיר חכם או מחשב נייד.	לקוח	לא פונקציונלית	5
3.1.4	ניהול משתמשים	הוספת משתמש	המשתמש חייב להגדיר העדפה/ות של קטגוריה/ות קופון/ים שהוא אוהב.	לקוח	פונקציונלית	3
3.2.1	ניהול משתמשים	הוספת בית עסק	מנהלי מערכת יוסיפו בעלי עסקים למערכת.	לקוח	פונקציונלית	5
3.2.2	ניהול משתמשים	הוספת בית עסק	המערכת תתמוך בבעל עסק אחד לכל בית עסק.	מנתח מערכות	לא פונקציונלית	3
3.3.1	ניהול משתמשים	התחברות משתמש	המשתמש יוכל להתחבר למערכת באמצעות פרטי זהויה שהזין בעת הרשמתו.	לקוח	פונקציונלית	5
3.3.2	ניהול משתמשים	התחברות משתמש	במידה והמשתמש אינו זוכר את הסיסמא, המערכת תשלח לו סיסמא חדשה באמצע לקוח.	פונקציונלית	פונקציונלית	3
3.3.3	ניהול משתמשים	התחברות משתמש	בעת התחברות המשתמש, המערכת תאגור את פרטי המיקום הנוכחי שלו.	מנתח מערכות	פונקציונלית	4
4.1.1	ניהול הזמנות	הזמנת קופון	המערכת תתמוך בהזמנת קופונים מבתי עסק ע"י משתמשים רשומים במערכת.	לקוח	מפוצל	5
4.1.1.1	ניהול הזמנות	הזמנת קופון	המערכת תתמוך בהזמנת קופונים מבתי עסק.	מנתח מערכות	פונקציונלית	5
4.1.1.2	ניהול הזמנות	הזמנת קופון	הזמנת קופונים תתאפשר ע"י משתמשים רשומים במערכת.	מנתח מערכות	לא פונקציונלית	5
4.1.2	ניהול הזמנות	הזמנת קופון	משתמש אשר מחובר למערכת, יוכל לבצע הזמנה של קופון עד לתאריך המימוש האחרון של הקופון.	לקוח	לא פונקציונלית	5
4.1.3	ניהול הזמנות	הזמנת קופון	תהליך הזמנת הקופון יתבצע באופן מאובטח.	מנתח מערכות	לא פונקציונלית	4
4.1.4	ניהול הזמנות	הזמנת קופון	לאחר הזמנת הקופון וביצוע תשלום על ההזמנה, יקבל המשתמש קוד מיוחד (serial key) וקבלה למייל המאשרת את הזמנתו.	מנתח מערכות	פונקציונלית	4
4.2.1	ניהול הזמנות	מימוש קופון	בעת הגעת המשתמש לבית העסק למימוש הקופון, יזין בית העסק את הקוד שקיבל המשתמש לשם זהויה ואישור סופי של מימוש הקופון.	פונקציונלית	פונקציונלית	4
4.2.2	ניהול הזמנות	מימוש קופון	לאחר הזנת קוד הזהויה של הקופון ע"י בית העסק, יתעדכן הקופון למצב "מומש".	מנתח מערכות	פונקציונלית	4
4.3.1	ניהול הזמנות	דירוג קופון	משתמש שמימש קופון יוכל לדרגו (1-5) בכל עת ובכך להבטיח את אמיתות הקופון ובית העסק.	לקוח	פונקציונלית	2
4.4.1	ניהול הזמנות	צפייה בפרטי הזמנות	לאחר הזמנת הקופון, יוכל המשתמש לצפות בקופון/ים שהזמין.	לקוח	פונקציונלית	4
5.1.1	ניהול התראות	התראה על קופון	אם המנגנון מכון למצב התראה מיקום, המשתמש יקבל התראה על קופון הנמצא בקרבתו.	לקוח	פונקציונלית	5
5.1.2	ניהול התראות	התראה על קופון	המשתמש יקבל התראה על קופון לפי חישן ה-GPS הנמצא במכשירו.	לקוח	לא פונקציונלית	4
5.1.3	ניהול התראות	התראה על קופון	אם המנגנון מכון למצב העדפה, המשתמש יקבל התראה על קופון לפי קטגוריות שהזין בעת הרשמתו למערכת.	לקוח	פונקציונלית	4
5.1.4	ניהול התראות	התראה על קופון	המשתמש יקבל התראה על קופון אחת ל-10 דק'.	לקוח	פונקציונלית	5
5.1.5	ניהול התראות	התראה על קופון	אם המנגנון מכון למצב משולב, המשתמש יקבל התראה על קופון לפי קטגוריות שהזין בעת הרשמתו למערכת ובתנאי שמועד הקופון סמוך לשעה הנוכחית.	לקוח	פונקציונלית	4

אבירים אדירי 302991468
ליהיא ורצ'יק 308089333

ליטל מורלי 302491709
אורי בר-אילן 303013114

3	לא פונקציונלית	לקוח	הצגת ההתראה תהיה 10 שניות או עד שהמשתמש לוחץ "סגור" (הראשון מהשניים)	התראה על קופון	ניהול התראות	5.1.6
3	לא פונקציונלית	לקוח	לאחר שנספו 10 קופונים חדשים מרגע הפעלת המערכת, הם מוצגים למשתמש בתנאי שהמערכת אינה מציגה קופונים כעת המשתמש יקבל התראה מהמערכת על קופונים שרכש אך עדיין לא מימש, כשבוע לפני התאריך האחרון למימוש.	התראה על מימוש קרוב	ניהול התראות	5.2.1
3	פונקציונלית	לקוח	המשתמש יוכל לשנות את הגדרת מנגנון ההתראה.	שינוי הגדרות התראה	ניהול התראות	5.3.1
4	פונקציונלית	לקוח	המשתמש יוכל לכבות ולהדליק את מערכת ההתראות.	שינוי הגדרות התראה	ניהול התראות	5.3.2
3	לא פונקציונלית	לקוח	בעת מעבר ממצב כבוי למצב דלוק, המערכת תזכור את העדפת המשתמש מהפעם האחרונה שהיתה דלוקה.	שינוי הגדרות התראה	ניהול התראות	5.3.3
3	לא פונקציונלית	לקוח	לא ניתן להציג מעל ל-10 התראות קופונים ביום למשתמש	הגבלת התראות	ניהול התראות	5.4.1
3	לא פונקציונלית	לקוח	חיישן המיקום דוגם את מיקום המשתמש בכל שתי דקות.	שימוש בחיישנים	ניהול התראות	5.5.1
3	לא פונקציונלית	לקוח	חיישן הזמן דוגם ומעדכן את השעה בכל 15 דקות.	שימוש בחיישנים	ניהול התראות	5.5.2
5	פונקציונלית	לקוח	דו"ח קופונים של משתמש המשתמש יוכל לצפות בכל עת בקופונים שהזמין באמצעות המערכת.	דו"ח קופונים של משתמש	ניהול דוחות	6.1.1
5	פונקציונלית	לקוח	דו"ח קופונים של בית עסק יכולים לצפות בכל עת בהזמנות של הקופונים שלהם.	דו"ח קופונים של בית עסק	ניהול דוחות	6.2.1
3	פונקציונלית	לקוח	דו"ח קופונים של בית עסק יכולים לצפות בכל עת בדירוג הקופונים שלהם.	דו"ח קופונים של בית עסק	ניהול דוחות	6.3.1

5	מפוצל	מנתח מערכת	שמירת נתונים ותאפשר בצורה מרוחקת	שמירת נתונים מקוונת	שמירת נתונים	7.1.1
5	לא פונקציונלי	מנתח מערכת	נתוני המערכת יישמרו על חומרה שונה מזו של הלקוח	שמירת נתונים מקוונת	שמירת נתונים	7.1.1.1
4	פונקציונלי	מנתח מערכת	תוכנת הלקוח תיגש (תקרא ותכתוב) לנתונים בבסיס נתונים מרוחק	שמירת נתונים מקוונת	שמירת נתונים	7.1.1.2
3	לא פונקציונלית	מנתח מערכת	נתוני המערכת יישמרו בצורה שתייעל זמני חיפוש (יישום אינדקסים על חלק מהטבלאות)	ייעול שמירת נתונים	שמירת נתונים	7.2.1
3	לא פונקציונלי	מנתח מערכת	המערכת תתמוך בהוספת חיישנים	הוספת חיישן	ניהול חיישנים	8.1.1
3	פונקציונלי	לקוח	במקרה של בעיית תקשורת המערכת תבצע התחברות מחדש	בעיות תקשורת	ניהול חריגות	9.1.1
3	לא פונקציונלי	לקוח	המערכת לא תבצע יותר מ-5 ניסיונות התחברות רצופים.	בעיות תקשורת	ניהול חריגות	9.1.2
3	פונקציונלי	לקוח	המערכת תכבה את עצמה לאחר 5 ניסיונות התחברות כושלים רצופים.	בעיות תקשורת	ניהול חריגות	9.1.3

*in case the images are too small: The original requirements.xls
is in the repository

302991468 אבירם אדירי
308089333 ליהיא ורצ'יק

302491709 ליטל מורלי
303013114 אורי בר-אילן

Notes:

Git Repository:

All of the project's files (including the solution, pdf, word, diagrams and images are in our git repository)

<https://github.com/oriba/Analysis-Design-2>

Username: lolacoupons | Password: abcd1234

Unit-Testing:

1. Open:
NUnit: Coupons->NUnit-2.6.4->bin->nunit-x86.exe
2. Open tests:
File->open project->coupons->testUnit->bin->
debug->testUnit.dll
3. Run tests with the "Run" button.