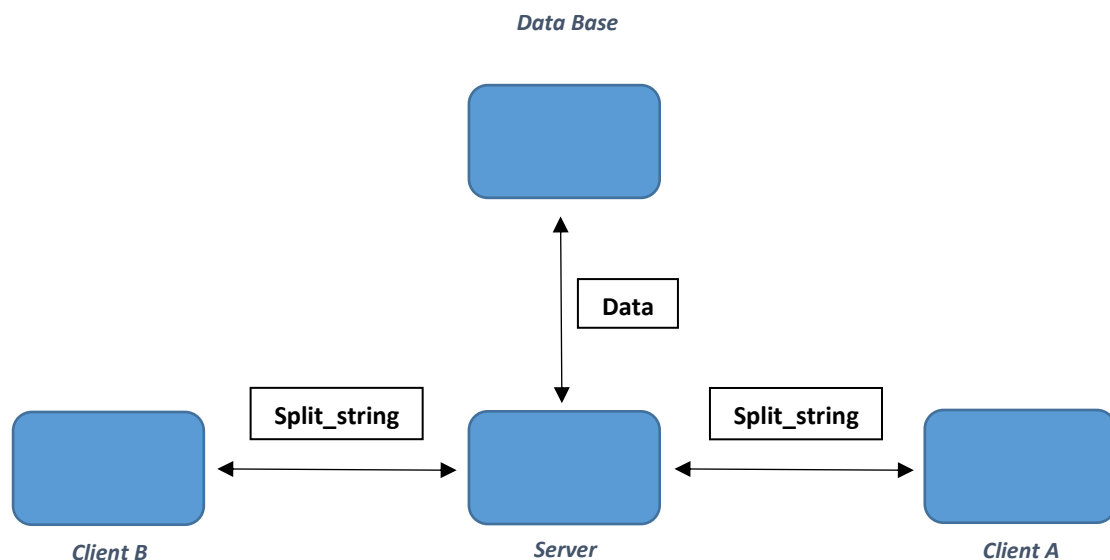


## DronePost – דואר רחפנים

ברוך הבא לפרויקט הגמר שלנו ב-JAVA! המימוש שלנו למערכת דואר הרחפנים עובד בצורה מלאה וע"פ הדרישות שניתנו בתרגיל. במסמך זה ננסה להסביר ולפשט את הכתוב בקוד ע"י פירוק המערכת והצגת החלקים וזרימת המידע, אז ללא כל עיכובים, בוא ונתחיל:

המערכת שלנו בנויה בבסיסה על פונקציונליות של 3 **virtual machines**: מחשב השרת ושני מחשבי לקוחות. (כאמור כל המימוש הוא קונספטואלי בלבד) על מנת שהמערכת המדומה תעבוד ישנו צורך לדמות את 3 **virtual machines** ע"י פתיחת 3 סביבות עבודה שונות ב-IDE של ECLIPSE. לאחר פתיחת 3 המכונות ומיקום הקבצים בתקיה הייעודית שלהם דרושה הרצה של 3 המכונות גם יחד על מנת להפעיל את המערכת!



דרך פעולת המערכת: 3 המחשבים מתקשרים אחד עם השני דרך שליחת **Strings** הבנויים בצורה הבאה: **String\_command-command\_info**. **String** עצמו מחולק ע"י הסימן "-" המחלק את המידע בין חלקי המערכת בכל אחד מהצדדים. בשני צדדי המערכת (צד שרת וצד לקוח) פונקציית ניהול המידע העיקרית בנויה כך שה**string\_command** יוביל את המערכת לתרחיש מסוים ולסנכרון פעולות בין שני הצדדים ואילו **command\_info** הוא המידע עצמו המנוהל בתוך המערכת. בנוסף כל מידע הנשלח בין מחשב הקליינט למחשב השרת הכולל בתוכו פעולות על אובייקטים נשלח ומגובה ב**DB** תו"כ תנועה.

**מחשב השרת:** כולל בתוכו מימוש של מספר מחלקות, חלקן מחלקות שירות וחלקן מחלקות המשמשות ליצירת אובייקטים, המחלקות כוללות: `system_control`, `Order`, `Drone`, `Subscriber`, `TimerDemo`, `connClass`, `Service`.

**System\_control:** מחלקה זאת אחראית על עיבוד המידע המתקבל ממחשבי הלקוחות. המחלקה כוללת פונקציה יחידה `Server_management()` המקצה חוט לכל משתמש חדש שמתחבר למערכת. ה-`Server_management()` בנויה בצורת `switch case` של `split_string[0]`. בהתאם לנמצא ב-`split_string[0]` המתקבל ע"י הקליינט, ייכנס החוט ל-`case` הרלוונטי ויבצע פעולות על אובייקטים ע"י שימוש במתודות שלהם ובשאר המידע המועבר בשאר התאים של `split_string[...]` בתור ארגומנטים למתודות הנ"ל. בנוסף, מחלקה זאת מחזיקה מספר מבני נתונים: 3 רשימות מקושרות הנועדו לייצג את המצב האפשרי של הרחפנים המשמשים למשלוחים, מערך המחזיק את כל המשתמשים שמחוברים ברגע זה למערכת וטבלת ערבול המחזיקה את כל ה-`sockets` של הקליינטים ע"פ ת.ז.

**Service:** מחלקת שירות הכוללת בתוכה מתודות המשמשות לעיבוד מידע ברמת התוכנה ומשומשות ע"י מחלקת `System_control`.

**connClass:** מחלקת שירות הכוללת בתוכה מתודות המשמשות לעיבוד ועדכון מידע ברמת ה-`DB` ומשומשות ע"י מחלקת `System_control`.

**TimerDemo:** מחלקת שירות האחראית על יצירת טיימר ל-`Drone`.

**Order:** מחלקה המשמשת ליצירת אובייקטים המייצגים את ההזמנות ברמת התוכנה.

**Drone:** מחלקה המשמשת ליצירת אובייקטים המייצגים את הרחפנים ברמת התוכנה.

**Subscriber:** מחלקה המשמשת ליצירת אובייקטים המייצגים את המשתמשים ברמת התוכנה.

**מחשב הקליינט:** כולל בתוכו מימוש של מספר מחלקות, מחלקה אחת ראשית (`Client_System`) המשמשת לעיבוד מידע, ושאר המחלקות המשמשות לפתיחת טפסים וקליטת מידע מהמשתמש, המחלקות כוללות בתוכן: `Form_address`, `Form_acceptance`, `Client_System`, `Form_my_orders`, `Form_Main`, `Form_lobby`, `Form_destination`, `Form_client_chat`, `Form_Sub`, `Form_payment`, `Form_Order`.

**Client\_System:** מחלקה זאת אחראית על עיבוד המידע המתקבל ממחשב השרת ובהתאם למידע להפעיל, לכבות ולעדכן טפסים המוצגים בצד לקוח. המחלקה כוללת פונקציה יחידה `client_management()`. בדומה לפונקציית ה-`Server_management()` בצד שרת, `client_management()` בנויה בצורת `switch case` של `split_string[0]`. בהתאם לנמצא ב-`split_string[0]` המתקבל מהשרת, ייכנס התהליך ל-`case` הרלוונטי ויבצע פעולות על טפסים בהתאם ל-`String_command` המתקבל ויעשה שימוש בשאר המידע המועבר בשאר התאים של `split_string[...]`.

שאר המחלקות בצד קליינט מייצגות טפסים למשיכת נתונים מהמשתמש. כאמור כל המידע נאסף מהצד קליינט ע"י מילוי טפסים מוגדרים, נשלח לצד שרת ומעובד לאובייקטים ועדכון המקומות הרלוונטיים ב-`DB`.

**Data Base:** בסיס הנתונים שאנו עובדים איתו הוא בסיס נתונים WAMP SERVER. בבסיס הנתונים בפועל קיימות שתי טבלאות המחזיקות את כל ה-DATA הרלוונטי: טבלת הזמנות וטבלת משתמשים. **לכל משתמש בטבלת המשתמשים יש גם שדה של מספר הזמנות אפשרי, שדה זה מתעדכן בהתאם לחבילת השירות שהמשתמש בחר לשלם ברישום לאפליקציה (חבילה עם מספר משלוחים קבוע או שירות חודשי).** אך ורק לצד השרת יש גישה ל-DB ואך ורק צד השרת אחראי לעדכן את ה-DB דרך מחלקת connClass האחראית על כל מה שקשור בהקמה, התחברות, משיכת מידע וניהול ה-DB.

### תיאור התרחישים במערכת

ראשית יש לפתוח את הצד שרת ב-virtual machine-1, לאחר שנפתח הצד שרת והשרת מחכה לקליינטים להתחבר, נפתח את שני הקליינטים ב-virtual machine-2 ו-virtual machine-3. (הקוד של שני הקליינטים זהה וניתן כתיקיה אחת).

כל קליינט הנפתח ב-virtual machine מתחבר בצורה אוטומטית לשרת ונפתח אצלו טופס כניסה **(נספח 1)**. ישנה אופציה להכנס בתור משתמש קיים או ליצור משתמש חדש.

אם נבחר ליצור משתמש נכנס לסדרת טפסים שיבקשו מהקליינט להכניס פרטים הרלוונטיים לתוכנה **(נספח 2-4)**.

אם נכנס למשתמש קיים ע"י הכנסת תעודת זהות וסיסמא, ייפתח טופס ה-lobby **(נספח 5)** אשר נותן לנו את האופציה לעבור למסך ההזמנות או לפתוח בצ"ט **(נספח 6)** עם משתמש מחובר.

אם נבחר בצ"ט ייפתח החלון הרלוונטי שבו נדרש להכניס ת.ז של המשתמש המחובר איתו נרצה לדבר ולהקליד את ההודעה. לאחר לחיצה על send ההודעה תשלח לשרת והשרת בתורו ישלח את ההודעה למשתמש הרלוונטי, אשר אצלו ייפתח חלון צ"ט בצורה אוטומטית.

אם נבחר במסך ההזמנות נוכל להכניס יעד הזמנה חדש **(נספח 7) (בצורת ת.ז של משתמש מחובר)** או להסתכל על כל ההזמנות שביצענו וקיבלנו במשתמש הנוכחי **(נספח 8)**. אם נבחר להכניס יעד הזמנה חדש, המערכת תשלח אלינו רחפן (קונספטואלית כמובן) אשר יגיע לאסוף את החבילה ויבקש מאתנו להזין כתובת יעד **(נספח 9)**. נוכל לבחור אם להשאיר את כתובת הנמען כמו שהיא מופיעה ב-DB או לשנות אותה לכתובת אחרת. ככה או ככה ברגע שנשלח את הרחפן עם החבילה, הרחפן יישלח לנמען ויופיע בצד הקליינט הנמען Form\_acceptance **(נספח 10)** המקבל את החבילה מהרחפן ומשחרר את הרחפן לדרכו, בנוסף השרת שולח הודעה לקליינט השולח שהחבילה שלו התקבלה ע"י הנמען.

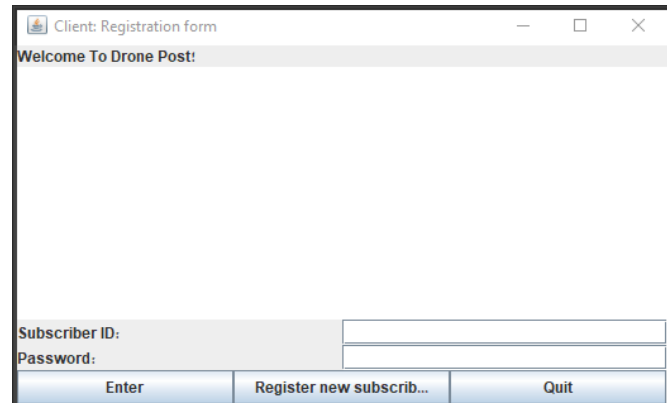
### פרטים נוספים

- על מנת שהמערכת תבצע את עבודתה כמצופה ראשית יש להתחבר מכל 3 הצדדים של המערכת ע"פ הסדר הכרונולוגי הבא! (א. צד שרת, ב. צד לקוח 1, ג. צד לקוח 2).

(כאמור הקוד המשומש לצד לקוח 1, זהה גם בצד לקוח 2)

- כנאמר כל המידע המועבר בין השרת-קליינט מועבר בצורת `split_string` ומתקבל בצורת `string-string-string...`. בשני הצדדים ישנה מתודה `split_string()` אשר יודעת לקבל את `String` הנ"ל ולחלק אותו לפי ה"-".
- מבחינה קונספטואלית כל משתמש חדש המתחבר לצד השרת מקבל הקצאת חוט משלו אשר מטפל בו בלבד. כלומר כל חוט מחזיק במספר משתנים משלו אשר מייצגים את פרטי הלקוח שקיבל את ההזמנה.
- כל הקוד עבר בדיקה של `meaningful names`. כל השמות של כל המשתנים והפונקציות מעידים על שימוש ומטרתם. כמובן שלכל אורך הקוד ישנן הערות אך הן שם אך ורק על מנת שלא יהיה צל של ספק במה מדובר. כלומר – לכל אורך קריאת הקוד, הקוד מדבר בעד עצמו.
- מבחינת הרחפנים : בתחילת הריצה של התוכנית ישנה יצירה של מספר קבוע של רחפנים אשר משמשים "להעביר" את החבילה ממשתמש אחד לשני, הדרך בה בחרנו לייצג את תנועת הרחפן היא כדלקמן : כאשר הצד שרת מקבל קריאה לרחפן מהצד קליינט הוא מוציא רחפן מראש הרשימה `available_drones`, מעמיס עליו את ההזמנה ומשנה אצלו את השדות הרלוונטים. לאחר מכן הרחפן מועבר לסוף הרשימה `busy_drones`. בכל סוף ריצה לאחר שלקוח היעד אישר את הקבלה של החבילה נשלף הרחפן הספציפי מרשימת `busy` לרשימת `available`. במידה ונגמר הטיימר לאחד הרחפנים במהלך המשימה (טכנית תרחיש זה אפשרי אך ורק במצב בו הנמען לא לוחץ על כפתור הקבלה או השולח לא מספר פרטי מען) נשלחת הודעה לשני הקליינטים המעורבים שהרחפן אבד ביחד עם החבילה והרחפן עצמו מועבר לרשימת `lost` ולא יוכל לשמש יותר למשלוחים.

## נספחים



Client: Registration form

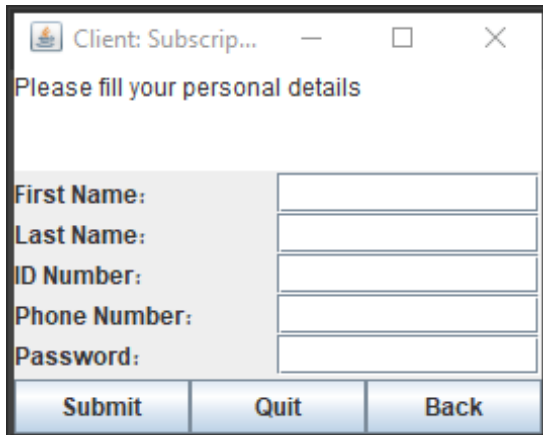
Welcome To Drone Post!

Subscriber ID:

Password:

Enter Register new subscrib... Quit

1. טופס כניסה



Client: Subscrip...

Please fill your personal details

First Name:

Last Name:

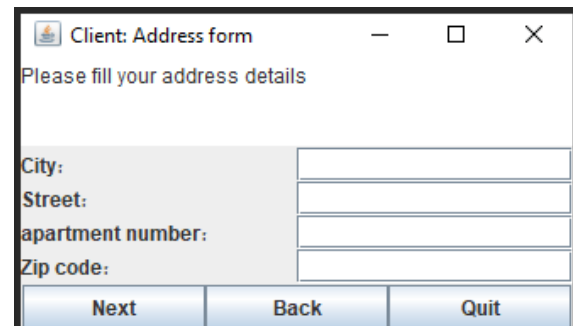
ID Number:

Phone Number:

Password:

Submit Quit Back

2. טופס פרטים אישיים



Client: Address form

Please fill your address details

City:

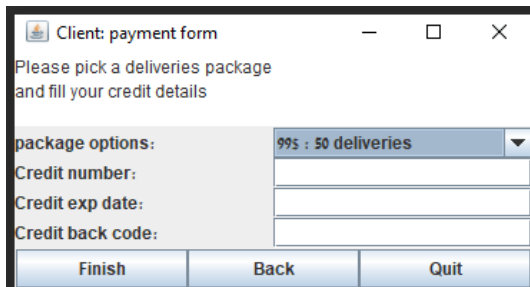
Street:

apartment number:

Zip code:

Next Back Quit

3. טופס פרטי כתובת



Client: payment form

Please pick a deliveries package and fill your credit details

package options: 99% : 50 deliveries

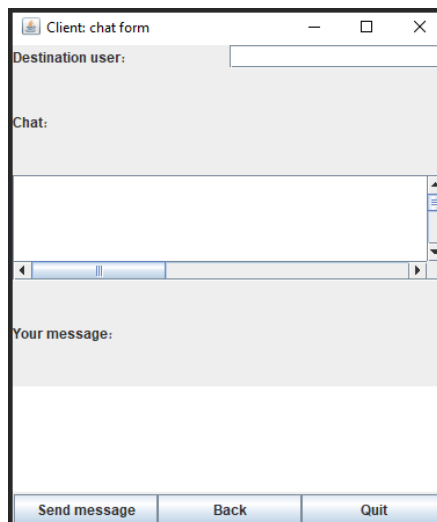
Credit number:

Credit exp date:

Credit back code:

Finish Back Quit

4. טופס פרטי תשלום



Client: chat form

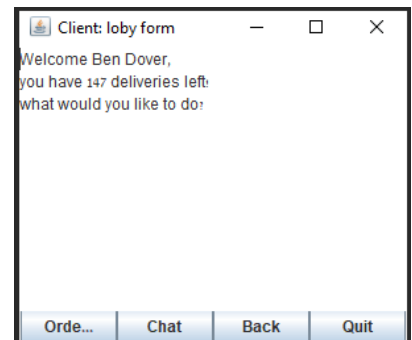
Destination user:

Chat:

Your message:

Send message Back Quit

6. טופס צ'ט

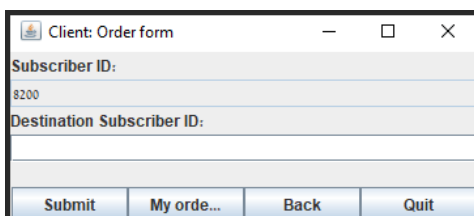


Client: lobby form

Welcome Ben Dover,  
you have 147 deliveries left:  
what would you like to do?

Orde... Chat Back Quit

5. טופס לובי



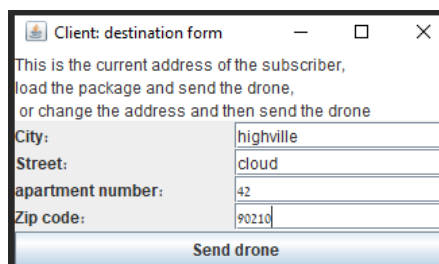
Client: Order form

Subscriber ID: 8200

Destination Subscriber ID:

Submit My orde... Back Quit

7. טופס רחפן



Client: destination form

This is the current address of the subscriber,  
load the package and send the drone,  
or change the address and then send the drone

City: highville

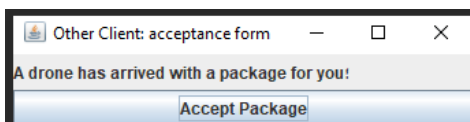
Street: cloud

apartment number: 42

Zip code: 90210

Send drone

9. טופס יעד

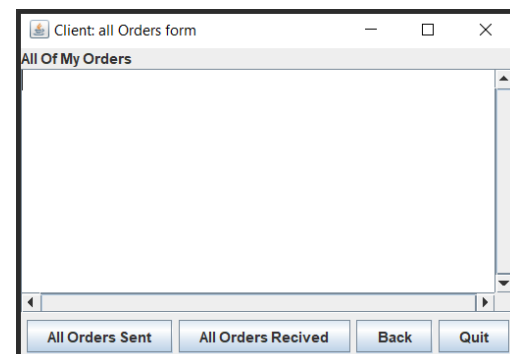


Other Client: acceptance form

A drone has arrived with a package for you!

Accept Package

10. טופס קבלה



Client: all Orders form

All Of My Orders

All Orders Sent All Orders Recived Back Quit

8. טופס הזמנות