**תבניות עיצוב - קבוצה 12  
  
1) Abstract Factory**  
בחרנו בתבנית עיצוב זו מכיוון שבפרוייקט שלנו אנו יוצרים אובייקטים תלויים, כולם בוחרים (voters) ואנו צריכים ממשק ליצירת אובייקטים אלו ללא ציון המחלקות הנוצרות. כך למעשה אנו שומרים על עיקרון הכימוס.  
המשתמש לא צריך לדעת איזה אובייקט נוצר בפועל כאשר מתבצעת כניסה למערכת.  
אם נרצה להוסיף מחלקות למערכת נעשה שינויים במחלקת FactoryVoter ולא בטופס Form1 (client). ההחלטה איזה אובייקט ליצור מתבצעת ב - FactoryVoter ולא בטופס Form1 .  
אופן היישום של תבנית העיצוב בקוד - ראשית ייצרנו ממשק בשם ICreateVoters שבה נמצאת המתודה createVoter.  
לאחר מכן יצרנו מחלקה בשם FactoryVoter שמממשת את הממשק ובמתודה createVoter יווצרו ויוחזרו אובייקטים בהתאם לנתונים המתקבלים. ב- Form1 נוצר אובייקט של FactoryVoter שמפעיל את המתודה הנ"ל.

**2) Command**  
בחרנו בתבנית עיצוב זו מכיוון שבפרוייקט שלנו קיימות 4 הרשאות, כאשר לכל הרשאה מותאם תפריט אופציות משלו. לשם כך אנו צריכים פקודה אחת OpenMenuשמשתמשת במתודה execute ומאפשרת פתיחה מסודרת של תפריטים בהתאם למשתמש הנכנס למערכת. כך למעשה בכל פעם שנרצה לפתוח תפריט חדש בהתאם להרשאות שיתווספו בספרינטים הבאים, נעשה זאת באמצעות מחלקת OpenMenu.  
אופן היישום של תבנית העיצוב בקוד -יצרנו ממשק הנקראICommand המצהיר על מתודה executeשבאמצעותה נפתח התפריט המתאים.  
קיימת מחלקה בשם OpenMenuשמממשת את הממשק והמתודה שמצויה בו.  
Form1 (client) יוצר אובייקט של OpenMenu ומריץ את המתודה דרך האובייקט.

**3) Singleton**

בפרוייקט שלנו החיבור ל-SQL Server הוא סינגלטון. מכיוון שמהלך כל הפרוייקט, ובמהלך הרצת המערכת יש שימוש רק בבסיס נתונים אחד ובחיבור יחיד אליו. לכן אין צורך ליצור אובייקטים מרובים של החיבור לבסיס הנתונים, כי הדבר גוזל משאבים.

אופן היישום של תבנית העיצוב בקוד:

ישנה מחלקה בשם SqlConnection ובה, בין השאר, בנאי פרטי- כדי שאף אחד מחוץ למחלקה לא יוכל ליצור אובייקט של המחלקה. שדה פרטי סטטי instance שמחזיק את האובייקט . ופונקצייה ציבורית סטטית Instance , שבודקת אם האובייקט כבר נוצר לפני כן, אם לא – תיצור חדש ותחזיר אותו, אם כן- תחזיר את הקיים. ולא תיצור חדש.