1. תרגול שימוש בתבניות (**Templates**), והתנסות בתהליך **Refactoring** בקוד קיים.
2. אהרון משה שמסיאן: 205779598, עקיבא משה סודרנסקי: 326382322

**3. הסבר כללי של התרגיל:**

התרגיל התמקד בשיפור קוד המייצג הזמנות קיימת באמצעות תהליך Refactoring ויישום עקרונות תכנות מונחה עצמים, תוך שימוש בתבניות (Templates). הקוד המקורי כלל חמישה סוגי טפסים להזמנות שונות – הזמנת מלון, השכרת רכב, הזמנת טיסה, רכישת כרטיס רכבת והזמנת כרטיס לאירוע – כאשר כל שדות הטפסים הוגדרו כמחרוזות (Strings), ללא הבחנה טיפוסית וללא ולידציה.

העבודה התחלקה לשני חלקים עיקריים:

1. **Refactor של קוד קיים** – שיפור המבנה של הקוד על ידי הפיכת השדות לגנריים באמצעות תבניות, צמצום כפילויות, והפרדת אחריות לפי עקרונות OOP. כל שדה הפך למחלקה עם טיפוס ייעודי, מה שאפשר קוד גמיש, קריא ותחזוקתי יותר.
2. **הוספת מנגנוני ולידציה** – הוספת ולידציה פנימית לכל שדה בטופס בהתאם לטיפוס הנתון (למשל, מספרים שלמים, תאריכים, מיילים), וכן ולידציה בין-שדות לאימות התאמה לוגית בין שדות שונים באותו טופס (למשל, תאריך כניסה ותאריך יציאה, שזמן היציאה אחרי זמן הכניסה). הקוד כוללת יכולת לזהות קלט שגוי ולהציג הודעות מתאימות למשתמש.

**שתי הקבצים שעליהם בחרנו לעשות את הנדרש הם**: **FlightBookingForm**, **HotelBookingForm**

**4. רשימות קבצי המשחק:**

**טפסים**

* **BookingForm**  
  מחלקת בסיס אבסטרקטית המייצגת טופס כללי להזמנה. אחראית על ניהול רשימת שדות, בדיקת תקינותם והצגת הטופס. משמשת כבסיס לטפסים קונקרטיים (כמו טיסות ומלונות).
* **FlightBookingForm**  
  מחלקה היורשת מ-BookingForm, מייצגת טופס להזמנת טיסות. מוסיפה שדות ייחודיים להזמנת טיסה כגון תאריך המראה, שדות לשדות תעופה ושעה מועדפת.
* **HotelBookingForm**  
  מחלקה היורשת מ-BookingForm, מייצגת טופס להזמנת מלונות. כוללת שדות כמו תאריכי צ'ק-אין וצ'ק-אאוט, מספר אורחים, סוג חדר ושם מלון.

**שדות בטופס**

* **AbstractField**  
  מחלקת בסיס אבסטרקטית לשדה בטופס. מגדירה את הממשק להצגת שדה, קבלת קלט מהמשתמש ואימות נתונים.
* **BaseField<T>**  
  שדה כללי (גנרי) שמטפל בסוג נתונים כלשהו (טמפלט). כולל לוגיקת ניהול ערכים, המרה למחרוזת, ואימות בסיסי. יורש מ-AbstractField.
* **StringField**  
  שדה המבוסס על BaseField<std::string>. מיועד לקלט טקסטואלי מהמשתמש, כולל אפשרות לאימות תווים.
* **IntField**  
  שדה המבוסס על BaseField<int>. מיועד לקלט מספרי שלם, כולל בדיקת תקינות (למשל טווחים או ערכים חיוביים בלבד).

**שדות ייעודיים**

* **Id**  
  שדה מזהה ייחודי לכל אובייקט בטופס. יורש מ-IntField וכולל לרוב דרישות פורמט מחמירות.
* **NamesField**  
  שדה בסיס לשמות. מוסיף פונקציונליות לטיפול בשמות פרטיים ושמות משפחה, כולל אימותים כמו התחלה באות גדולה.
* **Name**  
  מייצג שדה שם פרטי של המזמין או הנוסע. יורש מ-NamesField.
* **Address**  
  שדה קלט עבור כתובת מלאה, כולל רחוב, מספר בית ועיר. יורש מ-StringField.
* **Email**  
  שדה טקסט המיועד להזנת כתובת דוא"ל, כולל אימות מבוסס תבנית.
* **DepartureAirport**  
  מייצג את שם שדה התעופה שממנו תצא הטיסה. שדה טקסט עם אימותים מתאימים.
* **ArrivalAirport**  
  מייצג את שם שדה התעופה שאליו תגיע הטיסה.
* **Date**  
  שדה מחרוזת שמייצג תאריך כללי בפורמט טקסטואלי, למשל YYYY-MM-DD. מהווה בסיס לשדות תאריך נוספים.
* **DepartureDate**  
  מייצג את תאריך ההמראה בטופס טיסה. יורש מ-Date.
* **CheckInDate**  
  מייצג את תאריך הצ'ק-אין בטופס מלון.
* **CheckOutDate**  
  מייצג את תאריך הצ'ק-אאוט בטופס מלון.
* **SelectionOptions**  
  שדה טקסט שמייצג בחירה מרשימת אפשרויות (כמו drop-down), כולל אימות שהקלט נמצא ברשימה.
* **PreferredTime**  
  שדה לבחירת שעה מועדפת לטיסה מתוך רשימת זמנים מוגדרת מראש. יורש מ-SelectionOptions.
* **RoomType**  
  שדה לבחירת סוג חדר (יחיד, זוגי, סוויטה וכו'), בטופס המלון. יורש מ-SelectionOptions.
* **HotelName**  
  שדה טקסט לשם המלון אותו רוצים להזמין.
* **NumberGuests**  
  שדה מספרי שלם עבור מספר האורחים בהזמנה. יורש מ-IntField.

**רכיבי עיצוב/תצוגה בטופס**

* **Title**  
  מחלקה שמייצגת את כותרת הטופס, כולל שם סוג ההזמנה ("הזמנת טיסה", "הזמנת מלון" וכו'). מאפשרת הצגה בולטת בראש כל טופס.
* **FormButton**  
  מייצגת כפתור בטופס (למשל "Donc", "Cancl"). כוללת לוגיקת פעולה בהתאם לסוג הכפתור.

**5. פירוט מבנה נתונים עיקריים:**

* m\_fields – וקטור של מצביעים חכמים (unique\_ptr) לשדות הטופס, כל שדה יורש מ־AbstractField. השדות יכולים להיות מסוגים שונים (כגון טקסט, מספר, תאריך), והם מייצגים את שדות הקלט שהמשתמש ממלא בטופס.
* m\_selectionOptions – מערך של זוגות (<string, bool>) המייצגים אפשרויות לבחירה עבור שדות מסוג בחירה (למשל שעות מועדפות או סוגי חדרים), כאשר כל זוג מכיל תווית ודגל שמציין אם נבחרה.
* m\_titleLabel – אובייקט מסוג Title שמציג את כותרת הטופס בראש המסך, עם טקסט, גופן וגודל מותאם.
* m\_doneButton, m\_cancelButton – שני כפתורים גרפיים שמאפשרים למשתמש להשלים את תהליך ההזמנה או לבטל אותו. כל כפתור כולל תגובה לאירועים גרפיים והצגה מתאימה.
* window – הפניה לחלון מסוג sf::RenderWindow שבו מצויר ומנוהל הטופס. משמש להצגת הגרפיקה והקלט מהמשתמש.
* formManager – מצביע למחלקת DialogueManager, שאחראית על ניהול זרימת הדיאלוגים, מעבר בין טפסים והצגת תגובות בהתאם לפעולות המשתמש.

**6. אלגוריתמים חשובים שהשתמשנו בהם.**

עבור שדה תעודת הזהות הישראלית (Id), יושם אלגוריתם ייעודי לבדיקה פורמלית של תקינות המספר לפי כללי הלשכה המרכזית לסטטיסטיקה בישראל. האלגוריתם כולל חישוב סכום ספרות עם משקלים משתנים ובדיקת ספרת ביקורת, ומוודא שמספר תעודת הזהות שהוזן אכן חוקי לפי הכללים.

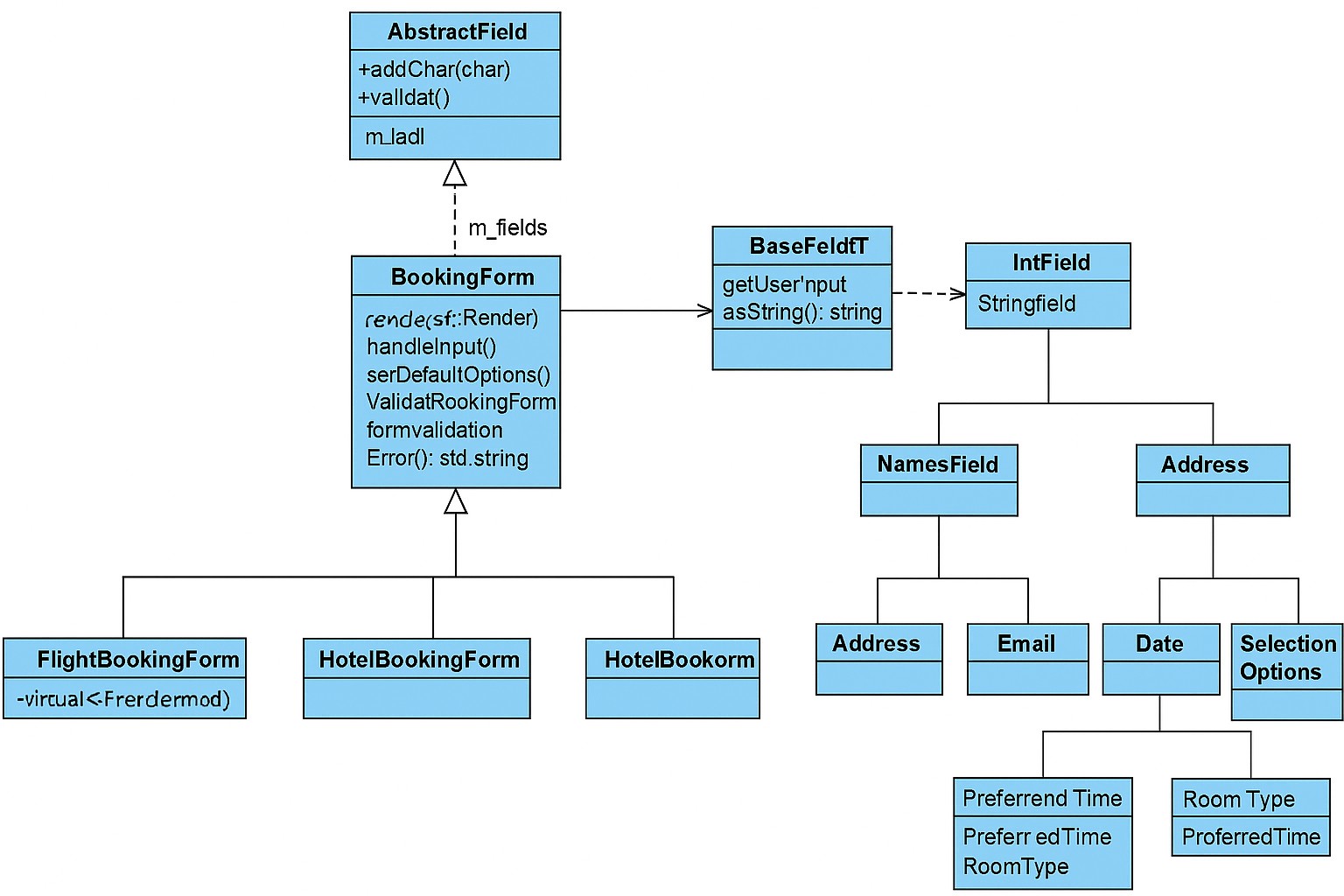
ועבור ייתר השדות בוצעה ולידציה לכל שדה בהתאם לטיפוס ולדרישות שלו. ברוב השדות (כגון שם, אימייל, תאריך, כתובת ועוד) בוצעה בדיקת תקינות בסיסית של קלט – לדוגמה, אימות שאינו ריק, שהקלט בפורמט תקין, או עומד בתחום ערכים שנדרשים.

**7. תיכון הפרויקט:**

הקוד עוצב מחדש במבנה מודולרי ומורכב, תוך יישום עקרונות תכנות מונחה עצמים (OOP), ביצוע Refactoring עמוק לקוד המקורי ושימוש מתקדם בטמפלייטים לצורך הכללה וגמישות. הקוד מבוסס על היררכיית שדות מודולרית, שבה **AbstractField** מייצגת שדה כללי בטופס, ומגדירה את הממשק להצגה, קלט ולידציה. **BaseField<T>** היא מחלקת טמפלייט המרחיבה את **AbstractField**, ומאפשרת יצירת שדות עבור טיפוסים שונים, תוך שמירה על ממשק אחיד. מחלקות קונקרטיות כמו **StringField**, **IntField**, **Id**, **Name**, **Email**, **Date** ועוד, יורשות מ-**BaseField<T>** ומייצגות שדות ייחודיים עם ולידציה ייעודית (למשל, **Id** כולל בדיקת ספרת ביקורת). שיטה זו מאפשרת להימנע משכפול קוד, ולהפריד בין לוגיקת תצוגה, ניהול נתונים ולידציה.

בנוסף, בוצע פיתוח של **BookingForm** כטופס כללי להזמנה, המכיל לוגיקה משותפת לכל הטפסים, כולל רשימת שדות, אפשרויות לבחירה, כפתורים, כותרת ולוגיקת הצגה. טפסים קונקרטיים כמו **FlightBookingForm** ו-**HotelBookingForm** יורשים מ-**BookingForm** ומממשים את השדות הרלוונטיים להם בלבד. השימוש ב-**BaseField<T>** מאפשר טיפול אחיד בשדות טופס, כך שהשדות נוצרו בצורה גנרית, מאפשרת הרחבה ולא דורשת הגדרה נפרדת לכל טיפוס נתונים.

לאור השכפול הרב בקוד המקורי, בוצע **Refactoring** מקיף, שכלל חילוץ שדות למחלקות עצמאיות עם אחריות ברורה, הפרדה בין קוד תצוגה, קלט ולידציה, והפיכת הקוד למבוסס טיפוסים באמצעות תבניות. כל השדות מיישמים ממשק אחיד, מה שמאפשר טיפול אחיד בטפסים השונים ובקוד כולה.

****

**8. אין באגים ידועים.**

**9. הערות כלליות:**

במהלך תיכון התרגיל חשבנו על דרך נוספת: שימוש במחלקה מרכזית אחת (Validator) שתאגד את כלל בדיקות התקינות (validations) של השדות. עם זאת, מאחר שהמוקד העיקרי של התרגיל היה עקרונות OOP והרחבה של מנגנון הולידציה, בחרנו בגישה שבה הפונקציה validate מוגדרת כוירטואלית טהורה במחלקת האב AbstractField. כל מחלקת שדה מממשת את הלוגיקה הספציפית שלה לבדיקה. (כמובן תוך התייעצות עם המרצה מתן פרל).

במקרים שבהם מספר שדות חולקים ולידציה זהה או חופפת, השתמשנו בהכלה – שדה אחד כולל אובייקט ולידציה פנימי משותף, ופונה אליו כחלק ממימוש הפונקציה validate. גישה זו אפשרה שמירה על עקרונות ההפרדה, הרחבה ותחזוקה עתידית נוחה יותר של הקוד.