בס"ד

REDME – OOP\_EX2: FlightBookingForm ו HotelBookingForm

צבי רולניק 214917965

. חנן גרשון כהן 213669229

3. כותרת התרגיל:

קיבלנו את התרגיל בתור קובץ שפותח עם כלי בינה מלאכותית התרגיל עוסק בהזמנת טפסים של מלון או של טיסה וכו תודה אנחנו הסטודנטים היינו צריכים לעשות רי פקטורינג על כל כלומר להוסיף מחלקות שנדרשות וכן הוספת וולדי טורים שהם בדיקות על השדות של הטופס כלומר לדוגמה יש שדה של שם המזמין אז אנחנו צריכים לבדוק שיש שם רק שם ועם מספרים לכתוב על המסך שיש כאן טעות.

4.

רשימת הקבצים שיצרנו:

4.1: **Butten – מחלקת כפתור גרפי**

מחלקת Butten מייצגת כפתור גרפי מותאם אישית לשימוש עם SFML.  
הכפתור מצויר כמלבן עם טקסט ממורכז, וניתן לבדוק האם המשתמש לחץ עליו בעזרת עכבר.

**תכונות עיקריות:**

* מציג טקסט במרכז הכפתור על רקע צבעוני.
* מאפשר ציור על חלון מסוג sf::RenderWindow.
* כולל פונקציית isClicked לזיהוי לחיצה באמצעות מיקום העכבר.

4.2:  
**Validator – מחלקת בסיס לאימות נתונים (Template)**

Validator היא מחלקת תבנית אבסטרקטית (template) שמספקת מבנה כללי לוולידציה של ערכים.  
היא משמשת כבסיס למחלקות נגזרות שמבצעות בדיקות חוקיות על ערכים מסוג כללי כלשהו (T), כמו מחרוזות, מספרים, תאריכים וכד'.

**מאפיינים עיקריים:**

* גנרית: תומכת בכל טיפוס נתון באמצעות תבניות (template<typename T>).
* אבסטרקטית: כוללת פונקציה וירטואלית טהורה isValid() שעל המחלקות היורשות לממש.
* מאפשרת הרחבה פשוטה לבדיקות שונות – לדוג' NameValidator, DateValidator וכו'.

4.3:  
**ChoiceFlight – בחירת טיסות**

מחלקה זו בודקת האם המשתמש בחר טיסה חוקית מתוך רשימת אפשרויות.  
היא יורשת מ־Validator<std::pair<std::string, std::vector<std::pair<std::string, bool>>>>, כאשר:

* first הוא מזהה היעד הנבחר,
* second היא רשימת זוגות של שם יעד ו־flag האם זמין.

**לוגיקה:**

* isValid() בודקת שהיעד הנבחר קיים ברשימת האפשרויות ושהוא זמין (true).

**🏨 ChoiceHotel – בחירת מלון**

מחלקה זו מטפלת בבחירת מלון מסוג מסוים מתוך רשימת מלונות חוקיים.  
היא יורשת מ־Validator<std::pair<std::string, std::vector<std::string>>>, כאשר:

* first הוא סוג החדר שנבחר,
* second היא רשימת סוגי חדרים זמינים.

**פונקציות:**

* isValid() – מאמת שהחדר הנבחר נמצא ברשימה.
* getRoomType() – מחזיר את סוג החדר שנבחר.

**📅 Date – תאריך**

מחלקת וולידציה לתאריך בפורמט תקני (כנראה dd/mm/yyyy).  
היא יורשת מ־Validator<std::string> ומוודאת שהתאריך:

* תקין תחבירית,
* תואם לתבנית הנדרשת,
* אינו תאריך בעבר (בהתאם למימוש הפונקציה getTodayDate()).

**פונקציות:**

* isValid() – מאמת את תקינות התאריך.
* getDate() – מחזיר את ערך התאריך.

**📧 Email – כתובת אימייל**

מחלקת Email אחראית לוודא שכתובת האימייל שהוזנה תקינה, על פי חוקים בסיסיים כמו:

* נוכחות @,
* תחום (domain) תקין,
* תווים מותרים.

היא יורשת מ־Validator<std::string>.

**🆔 Id – מספר תעודת זהות**

מחלקה זו בודקת חוקיות של תעודת זהות ישראלית לפי כללים כגון:

* אורך של 9 ספרות,
* סכום ספרות לפי אלגוריתם ביקורת (אם מיושם).

יורשת מ־Validator<int>.

**👤 Name – שם פרטי/משפחה**

מחלקת ולידציה לשמות משתמש, תוך בדיקה ש:

* השם אינו ריק,
* מכיל רק אותיות,
* עומד באורך סביר.

יורשת מ־Validator<std::string>.

**🛏️ RoomGuestMatcher – התאמת חדרים לאורחים**

מאמתת שהחדר הנבחר במלון מתאים למספר האורחים.  
יורשת מ־Validator<std::pair<ChoiceHotel, NumGuests>> ומכילה פונקציית match() פנימית שמבצעת את ההתאמה.

**⏳ StartEndDate – תאריכי התחלה וסיום**

מאמתת שרשימת תאריכים (בדרך כלל שניים – התחלה וסיום) תקינה:

* שניהם תאריכים חוקיים,
* תאריך הסיום מאוחר מהתאריך ההתחלתי.

יורשת מ־Validator<std::vector<Date>>.

**🧱 BaseField – שדה בסיסי מופשט**

מחלקה מופשטת (abstract) שמייצגת שדה כללי בטופס.  
מטרתה לאפשר עבודה פולימורפית עם כל סוגי השדות, בלי תלות בסוג הנתונים הפנימי.  
כוללת רק מתודה אחת טהורה:

cpp

CopyEdit

virtual bool isValid() const = 0;

כל מחלקה שיורשת ממנה חייבת לממש את הפונקציה הזו.

**🧩 Field<T> – שדה גנרי עם ולידציה**

מחלקה גנרית (template) שיורשת מ־BaseField ומכילה בתוכה אובייקט ולידציה מהטיפוס הנתון T.

**לוגיקה:**

* עוטפת אובייקט מסוג T (למשל: Email, Name, Date וכו').
* מיישמת את isValid() ע"י קריאה ל־m\_value.isValid().

כך ניתן לאחסן כל שדה בטופס בתוך אוסף של BaseField\*, ולבצע על כולם isValid() בלולאה אחת בלי לדעת את הסוג הספציפי.

BaseFlightHotel

מחלקה שמקשרת בין FlightBookingForm ו HotelBookingForm לבין BookingForm

ויש בה את הפונקציות ששותפות לשני המחלקות שאנחנו עובדים עליהם

5. מבני נתונים עיקריים ותפקידיהם .

std::vector<std::unique\_ptr<BaseField>> m\_fieldVec; ווקטור שבודק את כל הבעיות שבטופס מתחילת הווקטור עד מספר השדות שבטופס הנוכחי מאוחסנות כל הבדיקות על השדות עצמם וממקום זה עד הסוף הווקטור מאוחסנות הבדיקות ברמת הטופס

**: design 6. 🔷 מחלקת אב: BookingForm**

**מטרות:**

* מייצגת טופס כללי להזמנה (כל טופס ספציפי יורש ממנו).
* מכיל שדות כלליים כמו שם, ת"ז, כתובת ואימייל.
* אחראי על הצגת חלון אישור והצגת תוקף נתונים (validation).

**מרכיבים חשובים:**

* m\_fieldVec: וקטור של מצביעים חכמים לשדות (BaseField).
* fieldLabels ו-userInput: רשימות שמייצגות תוויות קלט וערכים מהמשתמש.
* פונקציה טהורה setDefaultValues() – מחייבת יורשים להגדיר ערכים ברירת מחדל.
* פונקציה analyzedVec() (וירטואלית) – יוצרת אובייקטי שדה לפי הקלט.
* checkValidate() – בודק תוקף לכל השדות.
* openConfirmationWindow() – מציג סיכום הנתונים בחלון נפרד.

**🔷 מחלקת יורשת: FlightBookingForm**

**מטרות:**

* מייצגת טופס להזמנת טיסה.
* מוסיפה שדות ספציפיים כמו שדה תאריך טיסה, שדה שעת טיסה, שדות שדות תעופה.

**הרחבות:**

* timeSelection: זוגות של שם שעה + האם נבחר כברירת מחדל.
* getVecTime() – מחזיר את וקטור השעות.
* analyzedVec() – יוצרת את כל שדות הטופס כולל ChoiceFlight.
* setDefaultValues() – מגדיר תאריך נוכחי כברירת מחדל, ושעת טיסה כ-"Don't Care".

**🔷 מחלקת יורשת: HotelBookingForm**

**מטרות:**

* מייצגת טופס להזמנת מלון.
* מוסיפה שדה בחירה לסוג חדר מתוך רשימה.

**הרחבות:**

* roomTypeSelection: וקטור של שמות החדרים.
* selectedRoomType: מצביע למיקום נבחר בוקטור.
* getArrayChoice() – מחזיר את הוקטור של אפשרויות החדר.
* analyzedVec() – יוצרת שדות ספציפיים להזמנת מלון (חסר במימוש שלך כרגע).
* m\_numOfFields = 11 – מציין מספר שדות.

**🧠 עקרונות OOP בולטים**

* **ירושה (Inheritance):** כל טופס (מלון, טיסה) יורש מ-BookingForm.
* **פולימורפיזם:** שימוש ב־virtual עבור פונקציות כמו render, handleInput, getFormType.
* **הפרדת תחומי אחריות (SRP):** BookingForm מטפל בתצוגת הטופס הכללית, והיורשים מוסיפים שדות ספציפיים.
* **שימוש ב־Composition:** באמצעות Field<T> שמקבל טיפוס של שדה (Name, Id, Date, וכו')

מחלקת "BUTTEN" היא מחלקה שעוסקת בכפתורים של התכנית . כך שיש לכל כפתור את הלוגיקה שלו שהמחלקה מטפלת בזה בעצמה.

9. הערות אחרות.

בבדיקת התאריך צריך שיהיה בצורה הבאה: YYYY-MM-DD בדווקא ולא כמו התאריך הדיפולטיבי שאינו בצורה הזו !