**Group Number 1 / Date: 01.6.16**

|  |  |  |
| --- | --- | --- |
| shayscal90@gmail.com | 302632105 | Shay Shahar |
| yakir1114@gmail.com | 312143969 | Yakir Karandian |
| rizraz2001@gmail.com | 305003659 | Raz Cohen |
| [asaff19899@gmail.com](mailto:asaff19899@gmail.com) | 200940997 | Asaf Tzarfati |

**Assignment 3 Answers:**

1. **שאלה:** תארו את תהליך השיקולים וקבלת ההחלטות שלכם בתכן( design )המערכת לטיפול בזימון תור לרופא מומחה עבור המטופל. תארו דילמות הנדסיות (issue Design )ספציפיות )לא כלליות( מתוך תכן המערכת שפיתחתם בהקשר זה )בהתאם להסבר ולדוגמאות בהרצאה(, והסבירו את הפתרונות שבחרתם.

**תשובה :**

במהלך תכנון הפרויקט בדקנו אפשרויות שונות למימוש זימון תור לרופא מומחה עבור מטופל. הדיון העיקרי היה כיצד נוכל לייצג תאריכים ושעות טיפול כיוון שתאריך הוא דבר אינסופי ולייצג אותו במסד נתונים זה בעייתי. אפשרות ראשונה להצגת התאריכים הייתה הזנת תאריכים ידנית ולאחר מכן חיפוש במסד הנתונים אם התאריך פנוי או לא, דבר אשר היה גורר חיפוש פגישה פנויה ארוך מהרגיל ולעיתים גם מייגע. ע"י השימוש שלנו בפרויקט ב-JavaFX החלטנו לייעל את תהליך בחירת התאריך ולהשתמש ב-Date Picker על מנת שיציג את לוח השנה עם התורים הפנויים והתפוסים ע"י כך שהאובייקט מתממשק ישירות עם מסד הנתונים. בסופו של דבר במסד הנתונים החלטנו להחזיק רק תורים תפוסים לכל רופא ותאריך בנפרד, בכך חסכנו את נושא יצירת התאריכים והחזקתם במערכת ויצרנו תהליך קביעת פגישה מהיר וידידותי למשתמש. בנוסף, ייצגנו רק את הפגישות התפוסות לרופא במסד הנתונים וכך כאשר המוקדן רוצה לקבוע תור למטופל בשעה מסוימת הוא מקבל את השעות הפנויות. כאשר רצינו לעבור בין מסכים נתקלנו בבעיה שלא הצלחנו לחזור לאותו מסך שממנו באנו. פתרנו את הבעיה הזאת ע"י מימוש Interface IUI אשר כל המסכים ממשים אותו. ולאחר מכן יצרנו מחסנית לכל המסכים שסוג המסך במחסנית יהיה מסוג IUI, וכך כל מסך שנכנסנו אליו אנחנו מכניסים למחסנית ואם יוצאים ממסך אז מוציאים אותו מהמחסנית וכך פתרנו את הבעיה של המסכים. בעיה נוספת שנתקלנו בה היא שכאשר שלחנו בקשה לשרת דרך Controller מסוים , לא הצלחנו לחזור לאותו Controller כאשר השרת החזיר לנו תגובה. פתרנו את הבעיה ע"י מימוש Interface Icontroller בControllers השונים בפרוייקט, ואז יצרנו משתנה סטטי מסוג Icontroller ששומר את ה-Controller הנוכחי שאנו נמצאים בו , וכאשר חוזרת התגובה מהשרת ל-HandleMessageFromServer אז אנחנו מפעילים על המשתנה של ה – Controller הנוכחי את פונקציית HandleReply ואז התוכנית יודעת לגשת לController המיועד.

1. **שאלה :** תארו את תהליכי הבדיקות שבצעתם במהלך פיתוח הפרויקט שלכם. ציינו את המרכיבים המאפיינים של התהליכים, בהתייחס לנלמד בהרצאות, ותוך מתן דוגמאות ספציפיות וקונקרטיות שביצעתם )או לא ביצעתם( בפועל במהלך הפרויקט )בהקשר של מרכיבי המערכת הספציפיים(

**תשובה :**

במהלך פיתוח הפרויקט תהליך הבדיקות שביצענו התנהל באופן הבא:

כאשר כתבנו את הקוד השתמשנו ב Glass Box Testing בשיטת Branch Coverage,

כך שכל אחד מחברי הצוות הריץ בדיקות על המודלים שהוא יצר , תיקן במידת הצורך את הנדרש לתקן וסיפר לשאר חברי הקבוצה על השגיאות והתיקונים כדי שיוכלו לאשר אף הם שהכול בסדר.

לרוב התגלו באגים קטנים , אך גם לעיתים רחוקות באגים גדולים שתוקנו ע"י מי שכתב את המודל או ע"י שאר חברי הקבוצה(מידה וחבר הצוות ביקש עזרה).

במהלך הבדיקות , הכנסנו קלטים שונים ומקרי קצה בכדי למדוד ולבדוק האם יש באגים או שגיאות ועל מנת לוודא שיוצא לנו הפלט המתאים והמצופה. כל אלה נבדקו בקפידה.

לדוגמה , כאשר משתמש מסוים מחובר ומנסה להתחבר שוב ע"י המערכת , האפשרות לא תינתן לו ותיזרק הודעת שגיאה בהתאם. הבדיקה לכך תעשה כך : כאשר המשתמש מתחבר משתנה הסטאטוס של ההתחברות שלו בDB ל 1 , וכאשר ינסה להתחבר שוב למערכת , המערכת תבדוק האם סטאטוס ההתחברות שלו כבר דולק , במידה וכן תזרוק הערה ולא תינתן לו האפשרות להתחבר , במידה ולא אז המשתמש לא מחובר – ותינתן לו האפשרות להתחבר.

החלק האחרון של הפרויקט נבדק על גבי מחשבי המכללה עם DB מעודכן ומפורט ובמידת הצורך נרשמו הערות ותיקונים סופיים.

1. **שאלה :** תחקור והפקת לקחים: התייחסו לאופן שבו התנהלתם לגבי 2 מרכיבים של הפרויקט: )א( שיתוף בין חברי הצוות בפיתוח וניהול גרסאות: ציינו יתרונות וחסרונות של השיטה שלפיה פעלתם )יש להתייחס לתהליך - לא להתמקד רק באספקטים טכניים(. )ב( שילובי קוד )לאחר הפיתוח - אינטגרציה( ובדיקות. אם היו קשיים מה הסיבה לכך? מה הייתם משנים בדיעבד?

**תשובה :**

א(

ההתייחסות לגבי שיתוף הפעולה בין ארבעת חברי הצוות בפיתוח וניהול גרסאות התבצע באופן הבא:

כאשר מתפרסמת מטלה , כל חברי הצוות נפגשים ומתקיים דיון הקובע את חילוק העבודה בצורה ברורה בין כל חברי הצוות וקביעת מועד סיום של העבודה(הנוכחית) עבור כולם.

בנוסף נפגשנו כל שבוע על מנת לבדוק התקדמות של העבודה , ובמידת הצורך עזרנו אחד לשני בעבודות השונות .

העבודה המשותפת והפגישות התקיימו לרוב פנים מול פנים , ולעיתים רחוקות , כאשר היה יותר לחץ היינו נפגשים און ליין ב Skype .

היתרונות הבולטים בשיטה זו :

-קביעת נהלים וחילוק העבודה בצורה שיטתית וברורה.

-עזרה והסברים למי שמעוניין(אחד מחברי הצוות) , במידת הצורך, ע"י שאר חברי הקבוצה.

-קביעת פגישה קבועה , כך שנפגשים ותעדכנים כל הזמן.

- נוחות.

החסרונות הבולטים בשיטה זו:

-פגישה במקומות רחוקים (=איבוד זמן).

ב)

ההתייחסות לגבי שילובי הקוד והבדיקות בין ארבעת חברי הצוות התבצעו באופן הבא:

סביבת העבודה בה השתמשנו הינה GitHub .

בתחילה היו קצת קשיים בלימוד השימוש ב GitHub , אך לאחר שימושים חוזרים התחלנו להבין את אופן השימוש בה.

כאשר כל חבר צוות ביצע שינוי או עדכון הוא הודיע על כך ב WhatsApp בכדי ליידע את שאר חברי הצוות על השינוי שנעשה.

לאחר עדכון הקוד , כל חבר צוות הריץ בדיקות על החלק עליו הוא אחראי , במידה והיו שגיאות או באגים ,חבר הצוות תיקן את הקוד או סיפר לשאר חברי הצוות וכולם נרתמו לפתור את הבעיה עד הפגישה הבאה.

בדיעבד , היה עדיף קודם כל ללמוד את אופן השימוש ב GitHub באופן שוטף ורק לאחר מכן להשתמש בה באופן גורף.