**Project sheets master**

1. Interface
2. Securety
3. Data base and tebales
4. users
5. Uplode excle and sheets files
6. Manege and merge files

משימות בעברית בכדי לזרז תהליכים:

1. אפיון
2. בניית הסביבה
3. תחילה פרונט react
4. באק python

**משימות:**

נשמע כמו פרויקט מעניין! הנה כמה המלצות לגביי טכנולוגיות, ספריות וכלים:

טכנולוגיות

**1. Frontend:**

- React: מתאימה לממשק אינטראקטיבי. מומלץ להשתמש ב-TypeScript כדי להבטיח טיפוסים חזקים.

- Material - UI (MUI): ספריית קומפוננטות שיכולה לשפר את חוויית המשתמש עם עיצובים מוכנים.

**2. Backend:**

- Python עם Flask או Django: שתיהן מצוינות לבניית APIs. Django כולל ORM מובנה, אך אם אתה מעדיף גמישות רבה יותר, Flask הוא בחירה טובה.

- FastAPI: אם אתה מחפש משהו מהיר ומתקדם, FastAPI מציע ביצועים טובים מאוד ותמיכה ב-Type hints.

**3. Database:**

- PostgreSQL: מסד נתונים מעולה עם תמיכה בטיפוסים מורכבים, מושלם עבור טבלאות גדולות.

- SQLAlchemy (ל - Flask) או Django ORM: לשימוש קל עם מסד הנתונים שלך.

**אבטחה**

- JWT: עבור אימות משתמשים.

- HTTPS: השתמש בהצפנה.

- CORS: הגדר את המדיניות בהתאם לצרכים שלך.

**וולידציה**

- Frontend: השתמש ב-yup או react - hook - form לוולידציה של טפסים.

- Backend: השתמש ב-Pydantic (עם FastAPI) או Django Forms לוולידציה בשרת.

תהליך עבודה

חלוקה לבלוקים:

**1. איפיון מערכת**

- הגדרת דרישות משתמשים.

- אפיון ממשק.

- תכנון מאגר נתונים.

**2. פיתוח Frontend**

- בניית קומפוננטות עם React.

- עיצוב ממשק עם MUI.

- אינטגרציה עם ה-API.

**3. פיתוח Backend**

- הגדרת מסד נתונים.

- בניית API.

- ניהול אבטחה.

**4. אבטחת איכות (QA)**

- בדיקות פונקציונליות.

- בדיקות ביצועים.

**5. הדרכה והשקה**

- הכשרת משתמשים.

- פרסום גרסה ראשונית.

פירוק למשימות בג'ירה

**1. איפיון מערכת:**

- איסוף דרישות

- יצירת דיאגרמת UML

- עיצוב Wireframes

**2. פיתוח Frontend:**

- יצירת קומפוננטת יבוא קבצים

- מיזוג נתונים

- סיכום חישובים

**3. פיתוח Backend:**

- יצירת API לפונקציות שונות

- הטמעת אבטחה

**4. בדיקות:**

- כתיבת בדיקות יחידה

- בדיקות אינטגרציה

**5. הדרכה והשקה:**

- הכנת מדריך למשתמש

- השקה והזנת משתמשים

אם יש שאלות נוספות או משהו ספציפי שתרצה להעמיק בו, אני כאן!

**העלאת האתר לאוויר:**

להעלאת האתר לאוויר עם תמיכה בעדכונים בזמן אמת דרך Git ויצירת דומיין, הנה תהליך מפורט:

**1. ניהול Git ופיתוח מתמשך**

- Continuous Integration (CI): הגדר תהליך CI אוטומטי עם GitHub Actions או CircleCI כדי להריץ בדיקות על כל `commit` שמוזרם ל-branch.

**2. עדכונים בזמן אמת עם Git**

- Git Hooks: ניתן להגדיר "Git Hooks" כדי לבצע פעולות אוטומטיות (כמו דחיפת עדכונים) כשמשהו משתנה ב-Git. לדוגמה, hook מסוג `post - receive` יכול לעדכן את האתר כאשר הקוד מתעדכן ב-branch מסוים.

- Webhook: כלי נוסף לניהול עדכונים בזמן אמת. ניתן להשתמש ב-Webhook שמחובר לשרת כדי לעדכן את האתר כאשר יש `push` ל-repo.

**3. שירותי Deployment**

- Heroku: פתרון ענן פשוט לפריסה, כולל אינטגרציה עם Git. כל `push` ל-branch יוביל לעדכון האפליקציה.

- Vercel או Netlify: שני שירותים שמתאימים מאוד לאפליקציות React, תומכים בעדכונים בזמן אמת עם Git ודחיפה אוטומטית של עדכונים כאשר יש commit חדש.

- Docker: יצירת image עם Docker יכולה לסייע בניהול סביבת הפיתוח והפריסה. הגדר קונטיינרים לפרויקטים שלך כדי לשמור על סביבת הפיתוח והפריסה זהה.

**4. יצירת דומיין**

- רכישת דומיין: תוכל לרכוש דומיין דרך ספקי דומיינים כמו GoDaddy או Namecheap.

- DNS Hosting: השתמש ב-CDN כמו Cloudflare לניהול DNS ולאבטחה משופרת. Cloudflare מציע הגנה ממתקפות DDoS ושירותי מטמון לשיפור ביצועים.

- SSL Certificate: הוסף תעודת SSL כדי להבטיח שהאתר יפעל ב-HTTPS, אפשר להשיג תעודה חינמית דרך Let’s Encrypt.

**5. פריסת Backend**

- DigitalOcean או AWS EC2: אם אתה בונה Backend, אלה פלטפורמות ענן שמאפשרות פריסה ישירה של API.

- Nginx: עבור ניהול בקשות HTTP, תוכל להשתמש ב-Nginx בתור proxy server לאפליקציה שלך כדי לנתב בקשות ולהבטיח פריסה יציבה.

- Docker Compose: אם אתה עובד עם Docker, השתמש ב-Docker Compose כדי להריץ כמה קונטיינרים (frontend, backend, database) בו זמנית.

**6. עדכונים אוטומטיים**

- Capistrano או PM2: תוכל להשתמש בכלים לניהול עדכונים. לדוגמה, PM2 מנהל אפליקציות Node. js ומאפשר עדכונים שוטפים בלי לעצור את האתר.

**7. מעקב לייב ואנליטיקה**

- Google Analytics: לחקר תנועת גולשים.

- Sentry: לניטור באגים ו-crashes בזמן אמת.

תהליך העלאה לאוויר ותחזוקה:

**1. הכנת הקוד:**

- וודא שהפרויקט מוכן לפריסה, כולל בנייה של קבצים סטטיים ב-React (כמו `npm run build`).

**2. העלאה לשרת:**

- העלה את הקבצים לשרת (Heroku, Vercel, וכו') או להריץ את הקונטיינרים אם אתה עובד עם Docker.

**3. תצורת דומיין:**

- הוסף את הדומיין דרך ספק הדומיינים שלך, עדכן את הגדרות ה-DNS אצל ספק הענן שלך.

**4. SSL ואבטחה:**

- התקן תעודת SSL והפעל הגדרות אבטחה נוספות.

**5. מעקב ועדכונים:**

- וודא שהעדכונים עוברים בזמן אמת עם Git hooks או Webhooks.

**6. תחזוקה ועדכונים שוטפים:**

- הרץ בדיקות אוטומטיות ב-CI.

- בצע עדכוני אבטחה ותיקוני באגים במערכת הניטור שלך.

אם תרצה להעמיק בכל אחד מהשלבים או לקבל דוגמאות ספציפיות, אני כאן לעזור!