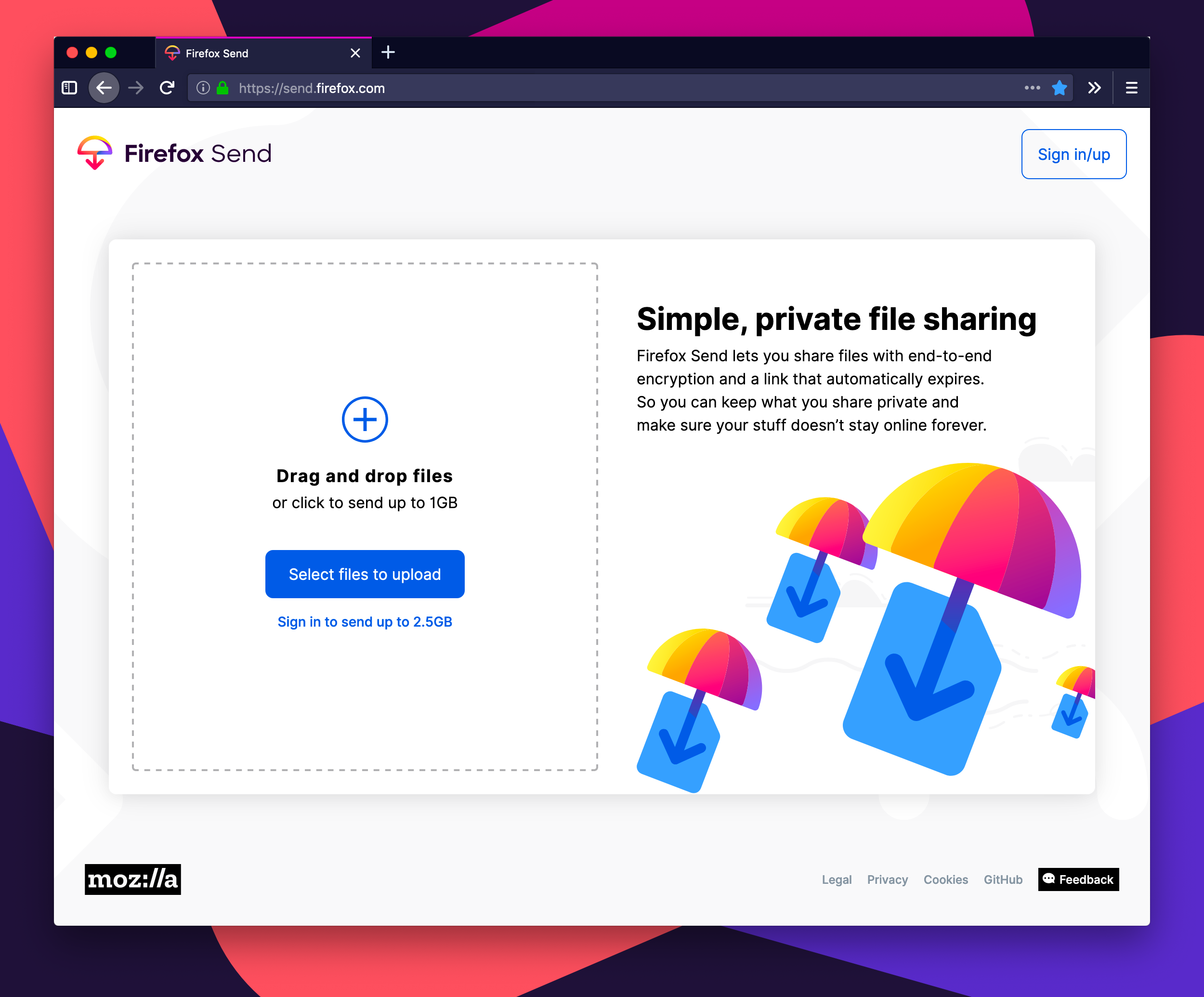
**Software Engineer, Backend (Python) Assignment**

ให้ผู้สมัครสร้างเว็บไซต์อัพโหลดไฟล์ คล้าย Firefox Send (send.firefox.com) ขึ้นมา ตามขั้นตอนต่อไปนี้

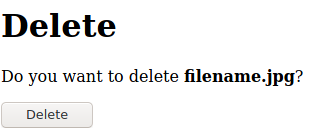
****

(ตัวอย่าง) เว็บอัพโหลดไฟล์ Firefox Send

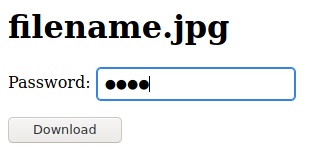
1. มี Project ให้สำหรับ Django ผู้สมัครจะใช้ Project ตามที่ให้หรือไม่ก็ได้ ([ดาวน์โหลดที่นี่](https://drive.google.com/file/d/1f03R7PVgF3gWdLyxNApq7XbuVxIwFSv2/view))
   * แนะนำให้ผู้สมัคร Upgrade Dependencies ต่างๆ ให้เป็นเวอร์ชั่นล่าสุดก่อนเริ่มทำงาน
   * กรณีต้องการสร้าง Project เอง สามารถเลือกใช้ได้ทั้ง Django หรือ Flask และต้องรองรับ Python 3.7 ขึ้นไป
2. หน้าอัพโหลดไฟล์ (ใน Project ตัวอย่าง ทำไว้ให้แล้ว)
   * ให้เลือกไฟล์ในเครื่องได้ทุกประเภท 1 ไฟล์
   * ผู้อัพโหลดจะต้องตั้งเงื่อนไข คือ จำนวนครั้งที่ดาวน์โหลดได้ (เลือกได้ตั้งแต่ 1 - 100 ครั้ง) และ เวลาเก็บไว้ (5 นาที - 7 วัน) เป็น Dropdown Box และต้องระบุทั้ง 2 อย่าง
   * ผู้อัพโหลดสามารถระบุรหัสผ่านหรือไม่ก็ได้
   * การตรวจ**ไม่คำนึงถึงความสวยงาม** หรือ User Experience (UX) ไม่จำเป็นต้องแก้ไข HTML เดิมที่ให้ไว้
   * **ไม่อนุญาต**ให้ใช้ CSS และ JavaScript library เช่น Bootstrap, Semantic UI, React หรือ jQuery



1. ทำ Backend ขึ้นมารับไฟล์ที่อัพโหลด
   * ตรวจสอบเงื่อนไขว่า ไฟล์ที่อัพโหลดขนาดไม่เกิน 100MB
   * ไฟล์ที่อัพโหลดแล้วให้เซฟลงบนเครื่อง
   * สามารถอัพโหลดไฟล์ชื่อซ้ำกันได้โดยไม่ทับกัน
   * ให้บันทึกข้อมูลที่จำเป็นลงใน SQLite Database หรือระบบเก็บข้อมูลอื่นๆ
   * เพื่อความง่ายในการตรวจ **ไม่อนุญาตให้โปรแกรมต่อ server อื่นๆ** เช่น MySQL, Redis, RabbitMQ
   * ผู้อัพโหลดจะได้รับลิงก์สำหรับดาวน์โหลด และ ลิงก์สำหรับลบไฟล์ (ผู้ที่ไม่มีลิงก์ จะไม่สามารถดาวน์โหลดไฟล์ได้ และเฉพาะผู้อัพโหลดเท่านั้นที่ลบไฟล์ได้)
2. ทำหน้าสำหรับลบไฟล์
   * แสดงปุ่มยึนยันว่าจะลบไฟล์ (ใน Project ตัวอย่าง ทำไว้ให้แล้ว)
   * เมื่อยึนยันแล้วให้ลบไฟล์ออกจาก Server



1. ทำหน้าสำหรับดาวน์โหลด (ใน Project ตัวอย่าง ทำไว้ให้แล้วเฉพาะ interface)
   * ตรวจสอบว่าไฟล์ที่จะดาวน์โหลดมีอยู่จริงและยังไม่หมดอายุตามเงื่อนไข
   * แสดงปุ่มดาวน์โหลด (ถ้าหากไฟล์มีการตั้งรหัสผ่าน ให้แสดงช่องกรอกรหัสผ่าน ถ้าไม่มีรหัสผ่านไม่ต้องแสดง)
   * เมื่อคลิกแล้วให้ตรวจสอบรหัสผ่าน (ถ้ามี) หากถูกต้องให้ Download File ทันที
   * ไฟล์ที่เซฟลงเครื่องจะต้องชื่อเดียวกับไฟล์ตอนที่อัพโหลด
   * ระบบควรสามารถรองรับผู้ใช้งานพร้อมกันได้จำนวนมาก



1. สร้างระบบลบข้อมูลอัตโนมัติ
   * เมื่อไฟล์หมดอายุตามเวลาที่ตั้งไว้แล้ว ระบบจะต้องลบไฟล์ออกเพื่อประหยัดพื้นที่ในการเก็บข้อมูล
   * การออกแบบขึ้นอยู่กับผู้สมัคร อาจจะแยกเป็นอีก Process หนึ่งไม่อยู่ใน Web Server ก็ได้
2. ทำ Dockerfile (ใน Project ตัวอย่าง ทำไว้ให้แล้ว)
   * เพื่อความปลอดภัย อนุญาตให้ใช้ Base Image จาก Docker Official Images เท่านั้น
   * Container ควรจะรันโดยไม่ใช้ Root และใช้ Layer เท่าที่จำเป็น
   * สามารถเลือกใช้ WSGI Server ได้ตามความเหมาะสม เช่น Gunicorn, uWSGI
   * ถ้าต้องการ Serve Static Files ใน Container ยังไม่ได้ตั้งค่าให้ Serve Static
3. ถ้าหาก Application ออกแบบไว้ให้ต้อง Start มากกว่า 1 Container ให้แก้ไขไฟล์ Docker Compose ที่ให้ไว้
   * Image อื่นๆ ที่ใช้ ต้องมาจาก Docker Official Images เท่านั้น (สามารถเขียน Dockerfile เพิ่มเติมเพื่อแก้ไข Image ได้ตามต้องการ)

## **วิธีการส่งงาน**

* ให้ zip source code ขึ้นไปตามลิงค์ที่กำหนดให้ โดยไม่ต้องส่งไฟล์ที่เอาไว้ใน VCS ignore มาด้วย (เช่น node\_modules, virtualenv)
* กำหนดส่ง**ภายใน 7 วัน**ตั้งแต่ได้รับโจทย์

## 

## **เกณฑ์การให้คะแนน**

การตรวจจะพิจารณาในหัวข้อดังนี้

* Function Requirement ครบถ้วน
* Security
* Code Quality และ Maintainability
* System Design
* Scalability