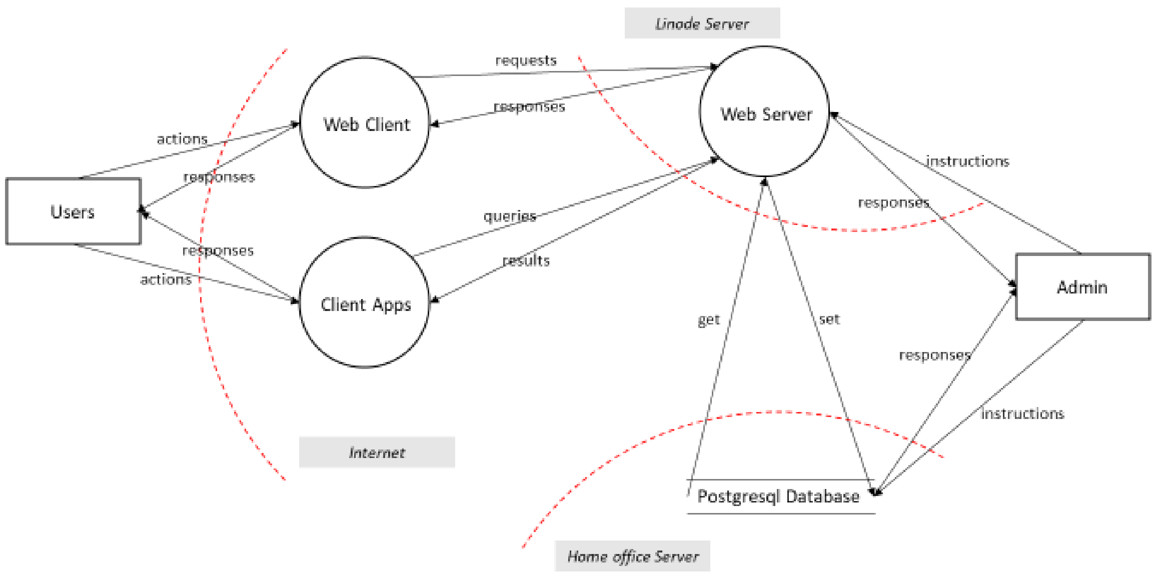
Name: Tianyi Lu



Threat: attackers create copy cat version of the client apps to access users login information.
Mitigation: download the client app from official websites or trusted app stores.
(spoofing)

Threat: attackers get user's credentials from other vulnerable servers.
Mitigation: Use two-factor authentication for logins from a different devices.
(Spoofing)

Threat: distributed denial of service attack on the database server.
Mitigation: Use multiple servers to balance the load. Use firewall to block malicious IP addresses from accessing the server.
(Denial of service)

Threat: attackers access APIs in high frequencies, slowing APIs' response time.
Mitigation: Use rate limiting on API calls from the same origin.
(Denial of service)

Threat: attacks use APIs on web server to access and modify any data.
Mitigation: Configure API keys to authenticate and authorize legitimate access. Have different levels of authorizations for APIs according to the data they access.
(Elevation of Privilege)

Threat: through other vulnerable services on the home office computer, attackers gain access to the database.
Mitigation: Use a dedicated server for hosting the database. Install firewall to block access on inactive

ports.
(Elevation of Privilege)

Threat: Vulnerabilities in PHP or PostgreSQL are used to gain access to the web server.
Mitigation: Update and patch all dependencies to the latest secure version regularly.
(Elevation of Privilege)

Threat: attackers use XSS to modify web client's content.
Mitigation: Use well-tested web frameworks to clean all user inputs.
(Tampering)

Threat: attackers implement SQL-injection on the web client to drop all tables in the database server.
Mitigation: Sanitize all user inputs before enter them into the database.

Threat: attackers launch man-in-the-middle attack between web client and web server.
Mitigation: User HTTPS for all interactions.
(Information Disclosure)

Threat: eavesdropper on user's network reads client app's HTTP-based API traffic with the server.
Mitigation: Use SSL/TLS encryption on all API traffic.
(Information Disclosure)

Threat: developers denied being responsible for writing bugs on the web server.
Mitigation: Keep an access log and change log on the web server. Assign different user account with the appropriate privileges to different developers.
(Repudiation)