SuperLap Racing Line Optimization System

EPI-USE

Quintessential

Amber Ann Werner [u21457752]

Milan Kruger [u04948123]

Qwinton Knocklein [u21669849]
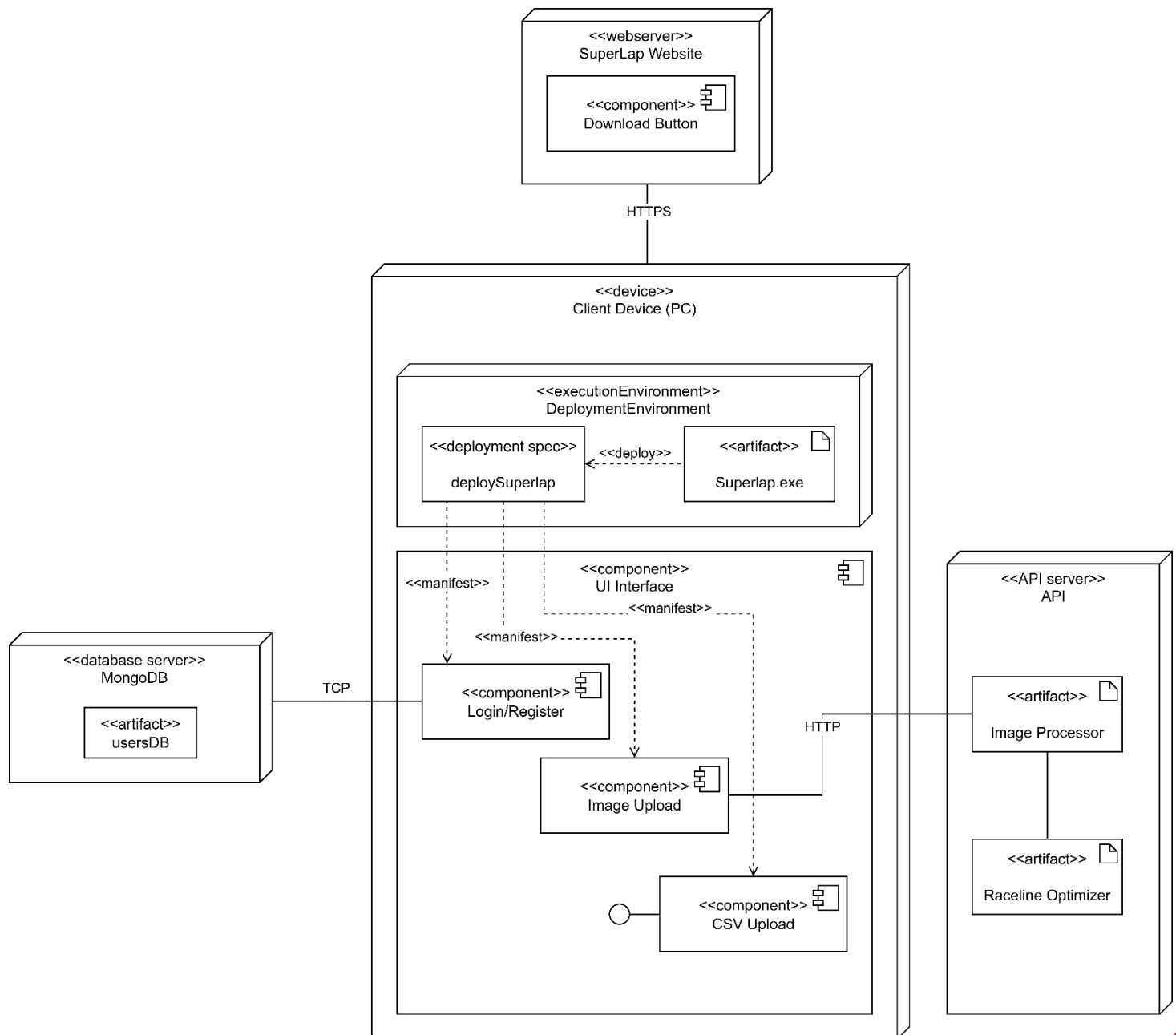
Sean van der Merwe [u22583387]

Simon van der Merwe [u04576617]

Deployment Model

Diagram

Target Environment

Our system is a desktop application that runs locally on the end-user's machine (Windows PC). The application must be downloaded from our official SuperLap Website and installed onto the client device.

- **Application type:** On-premises desktop application.

- Client requirements: Windows OS, internet access for updates and authentication.

- Supporting services:

  - Webserver: Hosts the SuperLap website and installation package (Superlap.exe).

  - Database Server: Centralized MongoDB database storing user information.

  - External API Service: Hosted independently from the client device, handling heavy-lift processing (e.g., Image Processor and Raceline Optimizer).

Deployment Topology

The system follows a multi-tier topology consisting of:

1. Presentation Tier (Client PC)

   - The user installs Superlap.exe on their PC.

   - The application provides a UI Interface with components such as:

     - Login/Register

     - Image Upload

     - CSV Upload

2. Data Tier (Database Server)

   - A remote MongoDB instance stores and manages user data (usersDB).

   - Communication with the client application occurs over TCP/IP.

3. Processing Tier (External API Service)

   - Deployed off the user's PC (e.g., on a cloud-hosted server).

   - Provides services such as:

- Image Processor

- Raceline Optimizer

o The client communicates with the API over HTTP.

o This offloading ensures performance, scalability, and maintainability.

Tools and Platforms Used

- **Website hosting:** Webserver supporting HTTPS downloads.
- **Database:** MongoDB (remote, accessible via TCP).
- **External APIs:** Deployed on a cloud provider or dedicated server.
- **Client Distribution:** Windows installer (Superlap.exe).

Quality Requirements Support

- **Scalability:** Processing is offloaded to an external API, enabling horizontal scaling without overloading the user's PC.
- **Reliability:** Separation of concerns ensures that if the client crashes, the backend services (API, DB) remain unaffected.
- **Maintainability:** External APIs can be updated independently without requiring users to reinstall the desktop application.
- **Security:** HTTPS and TCP connections secure communication between the client, webserver, API, and database.