# Virtual Stage Academy (VSA) Master Checklist

## 1. Infrastructure Setup

- [ ] \*\*GitHub\*\*  
 - [ ] Create a repository for VSA.  
 - [ ] Add a `.gitignore` file.  
 - [ ] Push initial local files (e.g., `TechHub` folder structure).  
 - [ ] Configure branch protection rules if needed.  
  
- [ ] \*\*Vercel\*\*  
 - [ ] Create a Vercel account (if not done already).  
 - [ ] Link Vercel to the GitHub repository.  
 - [ ] Deploy a test webhook endpoint (`hello\_world` app).  
 - [ ] Configure environment variables in Vercel.  
  
- [ ] \*\*Local Development\*\*  
 - [ ] Set up a local Python environment (e.g., `venv`).  
 - [ ] Install necessary dependencies (`Flask`, `requests`, etc.).  
 - [ ] Test a simple Flask application locally.

## 2. Zoom Webhook App

- [ ] Create a Zoom Developer Account.  
- [ ] Set up a new app in the \*\*Zoom Marketplace\*\*.  
 - [ ] Configure app details (name, description, etc.).  
 - [ ] Set up event subscriptions (e.g., `meeting.chat\_message\_sent`, `meeting.reaction\_added`).  
 - [ ] Add the webhook URL (from Vercel).  
- [ ] Test the webhook locally using tools like `ngrok`.  
- [ ] Deploy and test the webhook live.

## 3. Frontend Overlays

- [ ] Develop a basic \*\*HTML/CSS/JS\*\* overlay.  
 - [ ] Start with a chat overlay.  
 - [ ] Test dynamic data fetching via WebSocket or API.  
- [ ] Deploy overlays to Vercel and integrate with OBS.  
- [ ] Test overlays in a live presentation.

## 4. Backend Development

- [ ] Write scripts to handle incoming webhook data.  
 - [ ] Parse Zoom events (e.g., chat messages, reactions).  
 - [ ] Format data for frontend overlays.  
- [ ] Create a WebSocket server for real-time overlay updates.  
- [ ] Test backend scripts locally and deploy to Vercel.

## 5. Testing and Deployment

- [ ] Test integrations end-to-end:  
 - [ ] Chat data flows from Zoom -> Backend -> Overlay.  
 - [ ] Overlay displays correctly in OBS.  
- [ ] Conduct a mock live event to validate all components.