**🎯 LOPULLINEN AIKAJANA (YKSINKERTAINEN)**

📍 TÄNÄÄN (11.10):

Lue suunnitelma, päätä jatkatko

📍 VIIKKO 1-3 (11.10-31.10):

Remote-validointi + tekninen valmistelu

→ Työmäärä: 2h/ilta = 30h total

→ Harjoittelu = prioriteetti #1

📍 KAMPUS-SPRINT (3.11-23.11):

Guerrilla + lehtori-demo + beta-launch

→ Työmäärä: 10-20h/viikko

→ FULL FOCUS

📍 JOULU (joulukuu):

Kasvata beta-käyttäjiä

→ LOVe-tentit jatkuu

→ Word-of-mouth

**Ensimmäinen tulo: ~17.11 (beta-launch)** **Ensimmäinen maksava: ~24.11**

**💬 HENKILÖKOHTAINEN VIESTI**

Alkuperäinen suunnitelma oli:

"Validoi HUOMENNA kampuksella"

**Mutta et voi.** Harjoittelu estää.

**Tämä EI ole ongelma. Tämä on MAHDOLLISUUS.**

**3 viikkoa valmistautumisaikaa = ERINOMAISTA:**

✅ Rakennat teknisen puolen valmiiksi ✅ Remote-validointi (WhatsApp = riittävä signaali) ✅ Lehtori-kontakti ehtii vastata ✅ Kun kampus alkaa 3.11 → FULL SPRINT

**Harjoittelu sosiaalityössä = Bonus:**

* Verkostoidut (ehkä joku tekee LOVe-tenttiä?)
* Opettelet asiakaspuhetta (hyvä myynti-skill)

**Älä stressaa että et pääse kampukselle nyt.**

**Fokus:** Remote-validointi + tekninen valmistelu

**Kun 3.11 tulee → Olet 100% valmis.**

**Nyt: Päätä haluatko jatkaa.**

**Jos kyllä: Lähetä WhatsApp-viesti maanantaina 07:00.**

**Se riittää alkuun.** 🚀

Retry

J

Olisiko yksi hyvin tärkeä pointti mikä on unohtunut tämän sovelluksen verkkoon tuominen? Tällä hetkellä tämä toimii omalla tietokoneellani. Myös postresql konversio pitää tehdä ennen demoa. Riittääkö aika tähän? Osaatko antaa suosituksia minne julkaisen tämän tässä vaiheessa ja miten teen postresql konversion.

Strategized deployment, PostgreSQL migration, and hosting solutions.

17s

Vau, tämä on TODELLA kriittinen huomio! Käyttäjä on aivan oikeassa - koko sovellus pyörii vain lokaalisti hänen koneellaan. Tämä on MASSIIVINEN aukko suunnitelmassani.

Kriittiset asiat:

1. **Hosting/deployment** - Sovellus pitää saada verkkoon
2. **SQLite → PostgreSQL** - Tuotantotietokanta
3. **Domain** - loveenhanced.fi tai vastaava
4. **HTTPS** - Pakollinen turvalliseen käyttöön
5. **Environment variables** - Salaisuuksien hallinta
6. **Static files** - CSS, JS, kuvat
7. **Email-toiminnallisuus** - SMTP-asetukset

Tämä on TODELLA tärkeä ja minun olisi pitänyt huomata tämä aikaisemmin!

Arvioidaan aikaa:

* PostgreSQL migraatio: 2-4h (jos tietokanta on yksinkertainen)
* Hosting setup (Railway/Render): 2-3h
* Domain + DNS: 1h
* Environment variables setup: 1h
* Testing production: 2-3h TOTAL: 8-14h työtä

3 viikossa (harjoittelun aikana, illat): 2h/ilta × 15 iltaa = 30h → Riittää hyvin!

Suositukset hosting-palvelulle:

1. **Railway** (PARAS aloittelijalle)
2. **Render** (hyvä, ilmainen tier)
3. **Heroku** (kalliimpi, mutta helppokäyttöinen)
4. **DigitalOcean App Platform**

Tarkastelen eri hosting-vaihtoehtoja. DigitalOcean tarjoaa kehittyneemmän alustan, joka vaatii enemmän teknistä osaamista. Pilvipalvelun valinta riippuu projektin vaatimuksista ja omasta osaamistasosta.

PostgreSQL-migraatio näyttää suoraviivaiselta. Flask-SQLAlchemy helpottaa tietokantamuutoksia merkittävästi, ja connection stringin vaihto on kriittinen askel tuotantoympäristöön siirtymisessä.

**🚨 KRIITTINEN HUOMIO - Olet 100% OIKEASSA!**

**Tämä on MASSIIVINEN aukko suunnitelmassani!** Pahoittelen - olisin pitänyt huomata tämän heti.

**Nykytilanne:**

❌ Sovellus: localhost:5000 (vain sinun koneella)

❌ Tietokanta: SQLite (ei tuotantoon)

❌ Domain: Ei ole

❌ HTTPS: Ei ole

❌ Email: Toimiiko tuotannossa?

→ KUKAAN EI VOI KÄYTTÄÄ TÄTÄ

**Tämä on prioriteetti #1!** Ilman tätä, koko projekti on "koodi koneella".

**⏰ RIITTÄÄKÖ AIKA? (3 viikkoa = 30h iltoja)**

**Työarvio:**

| **Tehtävä** | **Aika** | **Prioriteetti** |
| --- | --- | --- |
| PostgreSQL-migraatio | 3-5h | 🔴 Kriittinen |
| Hosting-palvelun valinta & setup | 2-3h | 🔴 Kriittinen |
| Deployment (Railway/Render) | 3-4h | 🔴 Kriittinen |
| Domain + DNS | 1-2h | 🟡 Tärkeä |
| Environment variables | 1h | 🔴 Kriittinen |
| HTTPS/SSL | 0h (automaattinen) | ✅ Ilmainen |
| Email-toiminnallisuus (SendGrid) | 2-3h | 🟡 Tärkeä |
| Testing & bugit | 3-5h | 🔴 Kriittinen |
| **TOTAL** | **15-23h** | - |

**VASTAUS: Kyllä, aika RIITTÄÄ!**

3 viikkoa × 5 iltaa × 2h = 30h

Tarvitaan: 15-23h

Marginaali: 7-15h (bugeihin, yllätyksiin)

**🎯 SUOSITUS: Railway (Paras valinta sinulle)**

**Miksi Railway?**

| **Ominaisuus** | **Railway** | **Render** | **Heroku** | **DigitalOcean** |
| --- | --- | --- | --- | --- |
| **Hinta** | 5$/kk (start) | Ilmainen (rajoitukset) | 7$/kk min | 5$/kk |
| **PostgreSQL** | ✅ Sisäänrakennettu | ✅ Ilmainen | ❌ Kallis | ✅ Sisään |
| **Helppo deployment** | ⭐⭐⭐⭐⭐ | ⭐⭐⭐⭐ | ⭐⭐⭐⭐ | ⭐⭐⭐ |
| **Python-tuki** | ✅ Erinomainen | ✅ Hyvä | ✅ Hyvä | ⭐⭐⭐ |
| **Auto-deploy** | ✅ Git push | ✅ Git push | ✅ Git push | ⚠️ Manual |
| **Free tier** | ❌ Ei enää | ✅ Rajoitettu | ❌ Ei enää | ❌ Ei |

**Suositukseni: Railway**

* Helpoin setup
* PostgreSQL mukana
* Auto-deploy Git:stä
* Hyvä dokumentaatio
* 5$/kk = edullinen

**📋 TÄYDELLINEN DEPLOYMENT-OPAS (Askeleittain)**

**VAIHE 1: Valmistele koodi (2-3h)**

**A) Luo requirements.txt (jos ei ole)**

bash

*# Koneellasi, projektin juuressa:*

pip freeze > requirements.txt

**Tarkista että sisältää vähintään:**

Flask==3.0.0

Flask-SQLAlchemy==3.1.1

Flask-Login==0.6.3

psycopg2-binary==2.9.9 # PostgreSQL driver!

gunicorn==21.2.0 # Production server

python-dotenv==1.0.0

**Jos psycopg2-binary puuttuu:**

bash

pip install psycopg2-binary

pip freeze > requirements.txt

**B) Luo Procfile (Railway/Heroku tarvitsee)**

bash

*# Projektin juuressa, luo tiedosto: Procfile (ei .txt!)*

web: gunicorn app:app

**Mitä tämä tekee:**

* web: = Käynnistää web-serverin
* gunicorn = Production-tason WSGI server (ei flask run)
* app:app = app.py -tiedosto, app -muuttuja

**C) Luo runtime.txt (Python-versio)**

bash

*# Projektin juuressa, luo tiedosto: runtime.txt*

python-3.11.0

**Tarkista oma Python-versiosi:**

bash

python --version

*# Esim: Python 3.11.4 → Kirjoita runtime.txt: python-3.11.0*

**D) Päivitä app.py - Tietokanta-URL**

**ENNEN (SQLite):**

python

app.config['SQLALCHEMY\_DATABASE\_URI'] = 'sqlite:///love\_enhanced.db'

**JÄLKEEN (Environment variable):**

python

import os

*# Tuotannossa käytetään DATABASE\_URL, lokaalisti SQLite*

app.config['SQLALCHEMY\_DATABASE\_URI'] = os.environ.get(

'DATABASE\_URL', *# Railway asettaa tämän automaattisesti*

'sqlite:///love\_enhanced.db' *# Fallback lokaaliin kehitykseen*

)

*# Railway/Heroku käyttävät "postgres://" mutta SQLAlchemy haluaa "postgresql://"*

if app.config['SQLALCHEMY\_DATABASE\_URI'].startswith('postgres://'):

app.config['SQLALCHEMY\_DATABASE\_URI'] = app.config['SQLALCHEMY\_DATABASE\_URI'].replace('postgres://', 'postgresql://', 1)

**E) Luo .env -tiedosto (lokaalikehitys)**

bash

*# .env (ÄLKÄÄ lisätkö Git:iin!)*

DATABASE\_URL=sqlite:///love\_enhanced.db

SECRET\_KEY=your-secret-key-here

FLASK\_ENV=development

**Lisää .gitignore:**

.env

\*.db

\_\_pycache\_\_/

\*.pyc

venv/

**VAIHE 2: Railway Setup (30min)**

**1. Luo Railway-tili**

1. Mene: https://railway.app

2. Sign up with GitHub

3. Vahvista sähköposti

**2. Luo uusi projekti**

Railway Dashboard:

1. "New Project"

2. "Deploy from GitHub repo"

3. Valitse: LOVe-Enhanced repo

(Jos ei näy → "Configure GitHub App" → Anna pääsy repo:lle)

**3. Lisää PostgreSQL-tietokanta**

Railway Dashboard:

1. Projektin sisällä: "+ New"

2. "Database" → "Add PostgreSQL"

3. Railway luo automaattisesti tietokannan

4. Odota 1-2min että alustetaan

**4. Yhdistä PostgreSQL sovellukseen**

Railway tekee tämän AUTOMAATTISESTI!

Tarkista:

1. Klikkaa Flask-app

2. "Variables" -välilehti

3. Pitäisi näkyä: DATABASE\_URL = postgresql://...

Jos ei näy:

1. "Settings"

2. "Service Variables"

3. "Add Reference" → Valitse PostgreSQL

4. Reference Variable Name: DATABASE\_URL

**5. Aseta environment variables**

Railway → Flask-app → Variables → "+ New Variable"

SECRET\_KEY = [generoi vahva salaisuus]

FLASK\_ENV = production

**Generoi SECRET\_KEY:**

python

*# Python-konsolissa:*

import secrets

print(secrets.token\_hex(32))

*# Kopioi tulos → SECRET\_KEY*

**6. Deploy!**

Railway tekee tämän AUTOMAATTISESTI kun pusket Git:iin!

1. Commitoi muutokset:

git add .

git commit -m "Add production config"

git push origin main

2. Railway detektoi push:n

3. Rakentaa sovelluksen

4. Käynnistää automaattisesti

Seuraa: Railway Dashboard → "Deployments" → Katso logeja

**7. Hanki public URL**

Railway → Flask-app → "Settings"

→ "Generate Domain"

→ Saat: your-app.up.railway.app

Tallenna tämä URL!

**VAIHE 3: PostgreSQL-migraatio (2-4h)**

**A) Testaa yhteys lokaalisti PostgreSQL:ään**

**Railway → PostgreSQL → "Connect":**

Host: ...railway.app

Port: 5432

Database: railway

Username: postgres

Password: [kopioi]

**Testaa Python-konsolissa:**

python

import psycopg2

conn = psycopg2.connect(

host="...railway.app",

port=5432,

database="railway",

user="postgres",

password="..."

)

print("✅ Yhteys toimii!")

conn.close()

**B) Luo taulut tuotannossa**

**Vaihtoehto 1: Flask shell (Railway CLI):**

bash

*# Asenna Railway CLI:*

npm i -g @railway/cli

*# Kirjaudu:*

railway login

*# Linkitä projekti:*

railway link

*# Avaa shell:*

railway run python

**Pythonissa:**

python

from app import app, db

with app.app\_context():

db.create\_all()

print("✅ Taulut luotu!")

**Vaihtoehto 2: Init-skripti (helpompi):**

**Luo init\_db.py:**

python

from app import app, db

with app.app\_context():

print("🔄 Luodaan taulut...")

db.create\_all()

print("✅ Taulut luotu onnistuneesti!")

**Aja Railway:ssä:**

bash

railway run python init\_db.py

**C) Migratoi data SQLite → PostgreSQL**

**Jos sinulla on jo dataa (kysymykset yms):**

**Vaihtoehto 1: Export/Import (yksinkertainen):**

python

*# export\_questions.py (aja lokaalisti)*

import sqlite3

import json

conn = sqlite3.connect('love\_enhanced.db')

conn.row\_factory = sqlite3.Row

questions = conn.execute("SELECT \* FROM questions").fetchall()

questions\_list = [dict(row) for row in questions]

with open('questions\_export.json', 'w', encoding='utf-8') as f:

json.dump(questions\_list, f, ensure\_ascii=False, indent=2)

print(f"✅ Exportoitu {len(questions\_list)} kysymystä")

python

*# import\_questions.py (aja Railway:ssä)*

import json

from app import app, db

from models import Question *# Sinun malli*

with open('questions\_export.json', 'r', encoding='utf-8') as f:

questions\_data = json.load(f)

with app.app\_context():

for q\_data in questions\_data:

question = Question(\*\*q\_data)

db.session.add(question)

db.session.commit()

print(f"✅ Tuotu {len(questions\_data)} kysymystä")

**Vaihtoehto 2: Alembic (kehittyneempi, suositus pitkälle):**

bash

pip install Flask-Migrate

**Lisää app.py:**

python

from flask\_migrate import Migrate

migrate = Migrate(app, db)

**Aja:**

bash

*# Lokaalisti (SQLite):*

flask db init

flask db migrate -m "Initial migration"

*# Tuotannossa (PostgreSQL):*

railway run flask db upgrade

**VAIHE 4: Custom Domain (1-2h) [VALINNAINEN]**

**Osta domain (jos haluat)**

**Suositukset:**

* **Namecheap** (halpa, 10-15€/vuosi)
* **Cloudflare** (8-10€/vuosi)
* **Google Domains** (nyt Squarespace)

**Domain-ehdotukset:**

loveenhanced.fi

lovetentti.fi

love-app.fi

lovetentti.com

**Yhdistä Railway:hin**

1. Railway → Flask-app → "Settings" → "Domains"

2. "Custom Domain" → Syötä: loveenhanced.fi

3. Railway antaa DNS-asetukset:

Type: CNAME

Name: @

Value: your-app.up.railway.app

4. Mene domain-rekisteröijään (Namecheap/Cloudflare)

5. DNS-asetukset → Lisää CNAME-tietue

6. Odota 5-60min (DNS propagaatio)

7. ✅ https://loveenhanced.fi toimii!

**HTTPS:** Railway tekee automaattisesti (Let's Encrypt)

**VAIHE 5: Email-toiminnallisuus (2-3h)**

**Ongelma:** localhost SMTP ei toimi tuotannossa.

**Ratkaisu:** SendGrid (ilmainen 100 emailia/päivä)

**1. Luo SendGrid-tili**

1. https://sendgrid.com

2. Sign up (ilmainen tier)

3. Vahvista sähköposti

**2. Luo API-avain**

SendGrid Dashboard:

→ Settings → API Keys

→ "Create API Key"

→ Nimi: "LOVe Enhanced Production"

→ Permissions: Full Access

→ Kopioi API key (näkyy vain kerran!)

**3. Päivitä app.py**

**Asenna:**

bash

pip install sendgrid

pip freeze > requirements.txt

**Muuta email-funktio:**

python

*# ENNEN (SMTP - ei toimi tuotannossa):*

import smtplib

def send\_email(...):

with smtplib.SMTP('smtp.gmail.com', 587) as server:

*# ...*

*# JÄLKEEN (SendGrid - toimii tuotannossa):*

from sendgrid import SendGridAPIClient

from sendgrid.helpers.mail import Mail

import os

def send\_email(to\_email, subject, html\_content):

message = Mail(

from\_email='noreply@loveenhanced.fi', *# Vaihda omaan*

to\_emails=to\_email,

subject=subject,

html\_content=html\_content

)

try:

sg = SendGridAPIClient(os.environ.get('SENDGRID\_API\_KEY'))

response = sg.send(message)

return True

except Exception as e:

print(f"Email error: {e}")

return False

**4. Lisää environment variable**

Railway → Flask-app → Variables:

SENDGRID\_API\_KEY = [paste API key]

**VAIHE 6: Testing & Debug (2-3h)**

**Tarkistuslista:**

☐ Sovellus käynnistyy: https://your-app.up.railway.app

☐ Tietokanta toimii: Kirjautuminen onnistuu

☐ Rekisteröinti toimii

☐ Kysymykset näkyvät

☐ Tilastot toimivat

☐ Email lähtee (testaa salasanan palautus)

☐ HTTPS toimii (lukko selaimessa)

☐ Mobiilissa toimii

**Yleisimmät bugit:**

**1. "Internal Server Error"**

Railway → Deployments → Klikkaa viimeisin → "View Logs"

→ Lue error-viesti

→ Yleensä: typo, puuttuva env variable, tai DB-yhteys

**2. "Database does not exist"**

→ Et ajanut db.create\_all()

→ Aja: railway run python init\_db.py

**3. Static files ei lataudu**

# app.py:

app.static\_folder = 'static'

app.static\_url\_path = '/static'

# Varmista että Procfile:

web: gunicorn --bind 0.0.0.0:$PORT app:app

**4. Email ei lähetä**

→ Tarkista SENDGRID\_API\_KEY

→ Tarkista from\_email (pitää olla validoitu SendGrid:ssä)

**📅 PÄIVITETTY AIKATAULU (Deployment-fokus)**

**Viikko 1: 11.10-17.10**

**Maanantai 13.10:**

ILTA (19:00-21:00):

☐ Luo Railway-tili

☐ Asenna Railway CLI

☐ Luo Procfile, runtime.txt, requirements.txt

☐ Päivitä app.py (DATABASE\_URL)

**Tiistai 14.10:**

ILTA:

☐ Push Git:iin (jos ei vielä)

☐ Yhdistä Railway → GitHub repo

☐ Lisää PostgreSQL Railway:hin

☐ Käynnistä ensimmäinen deployment

**Keskiviikko 15.10:**

ILTA:

☐ Testaa: https://your-app.up.railway.app

☐ Debug jos ei toimi (lue logit)

☐ Aja db.create\_all() tuotannossa

**Torstai 16.10:**

ILTA:

☐ Migratoi kysymykset SQLite → PostgreSQL

☐ Testaa että kysymykset näkyvät

☐ Luo demo-tunnukset tuotantoon

**Perjantai 17.10:**

ILTA:

☐ SendGrid setup

☐ Testaa email-lähetys

☐ Final testing

☐ ✅ SOVELLUS VERKOSSA JA TOIMII!

**💰 KUSTANNUKSET (Budjetoi)**

| **Palvelu** | **Kustannus** | **Pakollinen?** |
| --- | --- | --- |
| **Railway** (Hobby tier) | 5$/kk | ✅ Kyllä |
| PostgreSQL (Railway:n sisällä) | 0$ (sisältyy) | ✅ Kyllä |
| **Domain** (Namecheap) | 12€/vuosi | ⚠️ Valinnainen |
| **SendGrid** (Free tier) | 0$ (100/day) | ✅ Kyllä |
| **Stripe** (maksutoiminto) | 1.4% + 0.25€ | ✅ Kyllä |
| **TOTAL (3kk)** | **~15-20€** | - |

**Ensimmäinen tulo:** 9.90€ × 10 käyttäjää = 99€ → Kattaa 5-6 kuukauden kulut heti!

**🚨 KRIITTISET MUISTUTUKSET**

**ÄLÄ unohda:**

❌ .env -tiedosto → GIT:iin

→ Lisää .gitignore:een!

❌ SECRET\_KEY kovakoodattuna

→ Käytä os.environ.get('SECRET\_KEY')

❌ DEBUG=True tuotannossa

→ app.config['DEBUG'] = False

❌ SQLite tuotannossa

→ PostgreSQL aina!

❌ flask run tuotannossa

→ gunicorn aina!

**✅ YKSINKERTAINEN CHECKLIST (Tee järjestyksessä)**

VIIKKO 1:

☐ Railway-tili luotu

☐ PostgreSQL lisätty

☐ Procfile, requirements.txt valmis

☐ app.py päivitetty (DATABASE\_URL)

☐ Push Git → Railway deploy automaattisesti

☐ db.create\_all() tuotannossa

☐ Kysymykset migratoidaan

VIIKKO 2:

☐ SendGrid setup

☐ Email-toiminnallisuus testattu

☐ Custom domain (jos haluat)

☐ HTTPS toimii

☐ Demo-tunnukset luotu tuotantoon

VIIKKO 3:

☐ Maksutoiminto (Stripe) asennettu

☐ Final testing: Käy läpi koko user flow

☐ Bugit korjattu

☐ ✅ VALMIS DEMOON!

**💡 VAIHTOEHDOT (Jos Railway ei toimi)**

**Plan B: Render**

Pros:

+ Ilmainen tier (rajoitukset: nukkuu 15min inaktiivisuuden jälkeen)

+ PostgreSQL ilmainen

+ Helppo käyttää

Cons:

- Hidas (ilmainen tier)

- 750h/kk limit

Hinta: 0€ (beta), 7$/kk (tuotanto)

**Plan C: PythonAnywhere**

Pros:

+ Helppo aloittelijoille

+ Hyvä dokumentaatio

Cons:

- Hidas

- Rajoitettu ilmaisversiossa

Hinta: 5$/kk (Hacker tier)

**Suositukseni:** Maksa Railway 5$/kk. Se on sen arvoista.

**🏁 LOPULLINEN VASTAUS KYSYMYKSEESI**

**"Riittääkö aika?"**

**KYLLÄ.** 15-23h työtä, sinulla 30h aikaa.

**"Minne julkaisen?"**

**Railway.** Helpoin + paras hinta/laatu.

**"Miten teen PostgreSQL-konversion?"**

**Seuraa yllä olevaa opasta askeleittain.**

**Deployment on PRIORITEETTI #1.**

**Ilman tätä, kaikki muu on turhaa.**

**Aloita tänään Railway-tilin luomisella.** 🚀